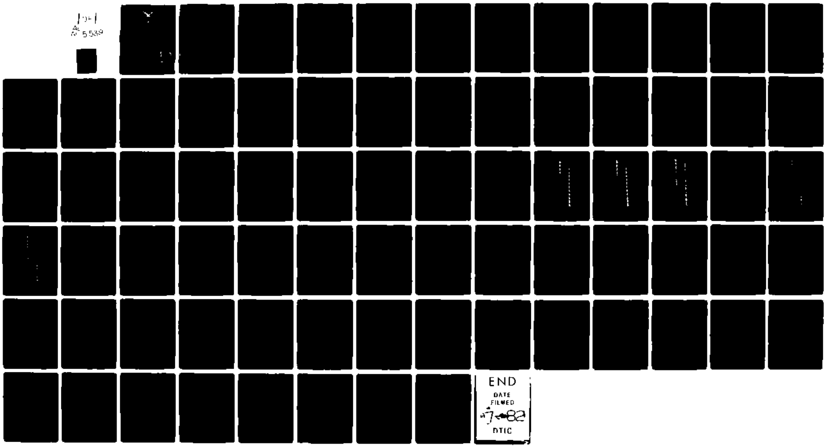
AD-A115 539 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 17/2
NOISE CANCELLATION TO IMPROVE THE QUALITY OF LPC PROCESSED SPEE--ETC(U)
DEC 81 C L BATCHELOR
UNCLASSIFIED AFIT/GE/EE/81D-10 NL



END
DATE
FILMED
7-82
DTIC

AD A115539

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

82 06 14 198

AFIT/GE/EE/81D-10

NOISE CANCELLATION TO IMPROVE THE QUALITY OF LPC PROCESSED SPEECH DEGRADED BY NOISE

THESIS

Christopher L. Batchelor
AFIT/GE/EE/81D-10 2Lt USAF

Approved for Public Release; Distribution Unlimited

AFIT/GE/EE/81D-10

Accession For	
NTIS GRA&I	X
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A	

DTIC COPY INSPECTED 2

NOISE CANCELLATION TO IMPROVE THE QUALITY OF LPC PROCESSED SPEECH DEGRADED BY NOISE

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science
in Electrical Engineering

by
Christopher L. Batchelor, B.S.E.E.
2d Lt USAF
Graduate Electrical Engineering

December 1981

Approved for Public Release; Distribution Unlimited

Preface

Linear predictive analysis and synthesis of speech is used as a basis for implementing low bit rate voice transmission and minimal digital storage of speech information. The synthesized speech resulting from the linear predictive analysis/synthesis of speech degraded by background noise is of poor quality. This report describes some of the methods proposed to improve the quality of this speech and describes the implementation and performance of one of these methods.

Thanks are due to my thesis advisor, Captain Larry Kizer, for overall guidance through this research. Also, Captain Kizer did a tremendous job ensuring that our Data General computers were maintained and operational. Professor Matthew Kabrisky was most helpful in providing insight into subjective effects of digital signal processing of speech. Lieutenant Robin Simmons is thanked for his assistance in the areas of computer system programming and understanding. Major Ken Castor was helpful in reviewing this report. Lastly, I wish to thank my typist (decoder), Ms. Sharon Gabriel, for doing a good job on this report.

Christopher L. Batchelor

Contents

List of Figures

Abstract

Methods for improving the quality of the speech resulting from linear predictive analysis/synthesis of speech degraded by background noise are discussed. A method of noise cancellation using Wiener filtering in the frequency domain with the short-time Fourier transform was chosen for implementation. Implementation was done on a Data General Nova/Eclipse digital signal processing system in FORTRAN 5. Speech degraded by white gaussian noise was processed through linear predictive analysis/synthesis with and without noise cancellation preprocessing. Preliminary laboratory listenings verified that an improvement in quality was achieved with noise cancellation preprocessing. Although improvement in quality was achieved, more effort is required to make this implementation more efficient and improve the quality of speech produced.

I. INTRODUCTION

Linear predictive coding (LPC) is a successful analysis/synthesis system for bandwidth compression of speech. Research indicates that LPC based systems degrade quickly when processing speech degraded by background noise (Ref 12:6). Thus, it is of interest to apply a digital noise cancellation technique to noise corrupted speech before LPC analysis/synthesis and then evaluate that technique's effectiveness in reducing degradation of quality.

Background

Linear prediction analysis is based on the idea that a speech sample can be approximated as a linear combination of past speech samples. Speech can be modeled as the output of a linear, time-varying system excited by either quasi-periodic pulses (voiced speech) or white noise (unvoiced speech) (Ref 13:38-106). Figure 1 is a block diagram of this model. The time-varying digital filter shown has the steady-state system function of the form:

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} \tag{1}$$

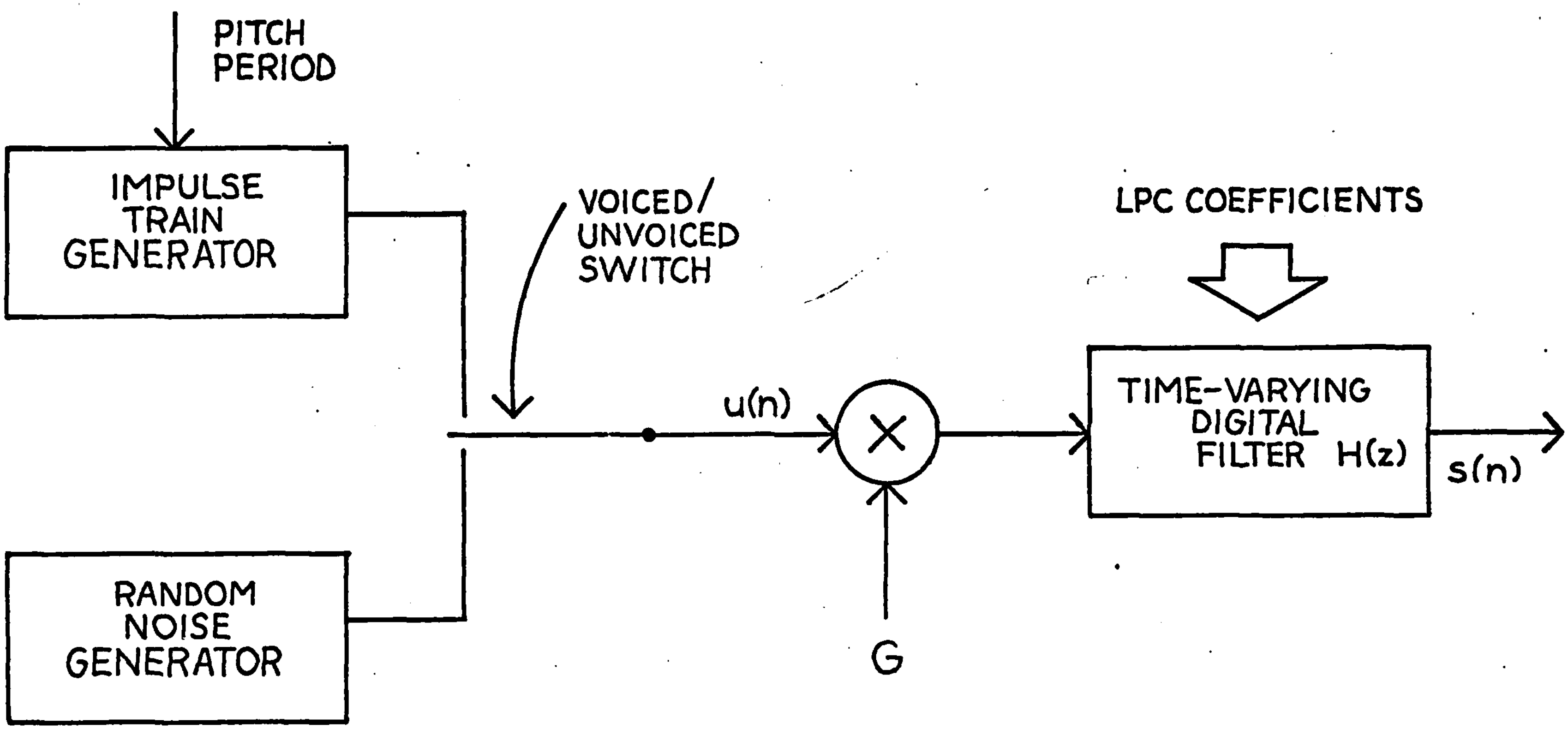


Figure 1. LPC Vocal Tract Model

This system function is an all-pole model and the poles define the resonances or formants of the model as determined by the coefficients $\{a_k\}$. The number p is the order of the model.

The application of linear predictive analysis to encoding speech for low bit rate transmission or storage is termed LPC (linear predictive coding). The LPC analysis parameters are the coefficients $\{a_k\}$, the gain parameter G, the pitch period, and a voiced-unvoiced parameter. Figure 2 shows a block diagram of an LPC vocoder. The transmitter codes the LPC analysis parameters for transmission through the channel and the receiver decodes the parameters and synthesizes the output speech. The LPC analysis parameters can be estimated by many different methods, as described in Digital Processing of Speech Signals by L.R. Schafer and R.W. Rabiner (Ref 13). These methods of estimation for the coefficients $\{a_k\}$ (which determine formants) are not noise tolerant.

Experimental research has demonstrated that four major differences exist between the all-pole linearly predicted spectra of clean and noisy speech (Ref 13:29-30). First, there is a loss of low energy formant information. Secondly, the formant frequencies are shifted. Thirdly, the bandwidth of each formant is wider. Lastly, an overall decrease of spectral dynamic range exists. If the signal to noise ratio is not too low, it has been observed that the primary perceptual effect is generation of "musical tone" like sounds in the background which causes degradation of speech quality

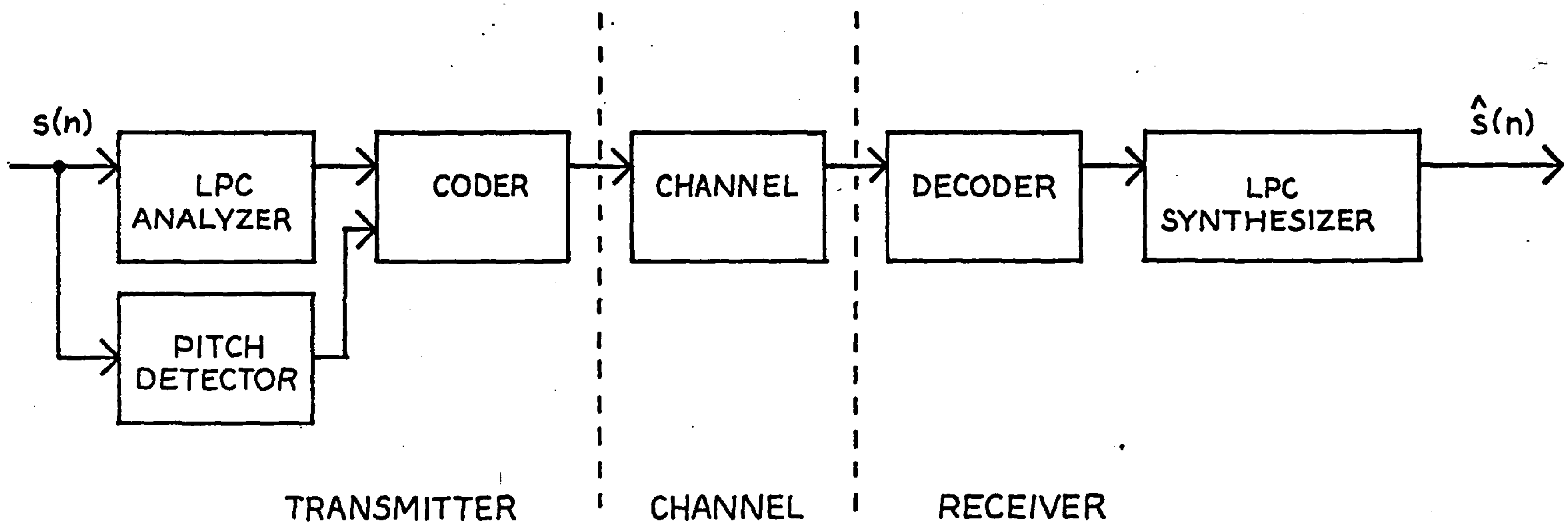


Figure 2. LPC Vocoder Model

(Ref 8:601). Apparently, estimation of the coefficients $\{a_k\}$ is not accurate due to the addition of noise.

Problem

The objective of this thesis is to examine methods to improve the estimation of the LPC coefficients of noisy speech and implement one of these methods on the Data General Nova-Eclipse speech processing system. After implementation, performance of the method was to be determined subjectively.

Scope

The methods to improve estimation of LPC coefficients to be examined will include noise cancellation pre-processing and noise tolerant estimation procedures. This thesis will not discuss estimation of the LPC coefficients using pole-zero modeling because preliminary results indicate that the increased complexity of pole-zero modeling does not improve performance of LPC (Ref 8)

Noise cancellation preprocessing was chosen over noise tolerant estimation procedures for implementation. A noise cancellation preprocessor can be used for other purposes to aid in robust speech processing.

Approach

Each method to improve estimation of coefficients will be described. Also, advantages and disadvantages of each method shall be discussed. Next, a detailed description of

the implementation of noise cancellation technique using short-time Fourier analysis will be given with accompanying results. Finally, recommendations for further research in this area shall be covered.

II. NOISE CANCELLATION METHODS

This section describes techniques which can be applied to improve LPC analysis/synthesis of speech corrupted with noise. The techniuqes described include time domain Wiener filtering, frequency domain linear prediction filtering, channel noise vocoder filter bank analysis filtering, linear prediction coefficient estimation using the Maximum A Posteriori (MAP) method, phase corrected spectral subtraction, and frequency domain Wiener filtering using the short-time Fourier transform. Unless otherwise specified, the following descriptions are summaries of the reference given.

Time Domain Wiener Filtering

This technique is the application of a Wiener linear prediction filter to reduce additive noise provided that the signal bandwidth is significantly less than the bandwidth of the additive noise (Ref 1). The filter would be applied as a preprocessor before LPC analysis/synthesis.

The implementation is in the time domain using the Widrow-Hoff LMS algorithm. Figure 3 is a schematic diagram of a finite impulse response Wiener filter. The coefficients $W^*(k)$, $(k=0,1,\ldots,L-1)$ are chosen to minimize the power in the error signal $e(j)$ and satisfy Eq (2). $\phi_{xx}(k)$ is defined as the autocorrelation of the input signal x. $\phi_{xy}(\ell)$ is the cross correlation between the input x and the output y.

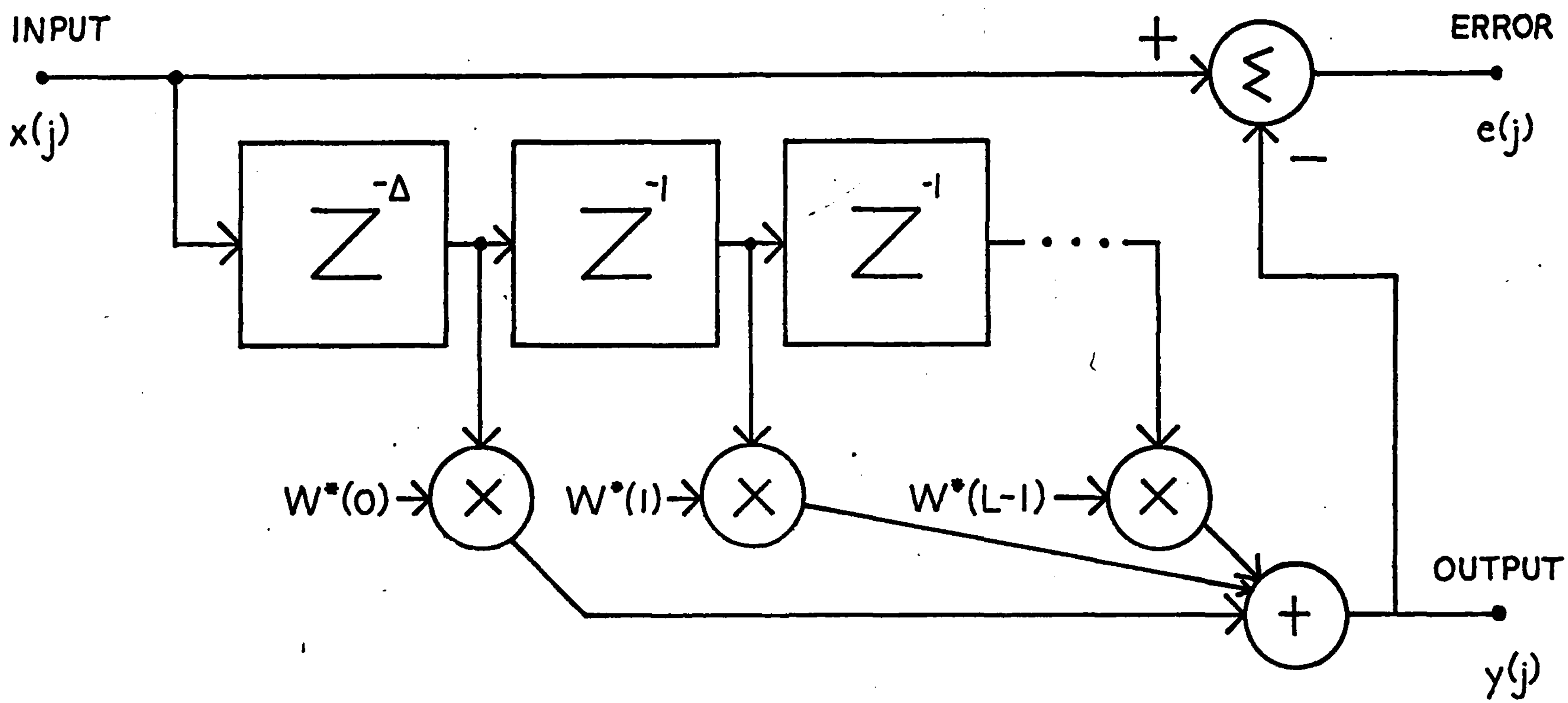


Figure 3. Finite Impulse Response Wiener Filter

$$\sum_{k=0}^{L-1} \phi_{xx}(\ell-k)W^*(k) = \phi_{xy}(\ell+\Delta) \qquad (2)$$

Noise suppression results from the fact that the decorrelation time for broadband noise is smaller than that for the narrowband signal. Therefore, it is possible to choose a value for Δ which will prevent the noise components from appearing in the output.

The advantages of this technique are that the prediction distance Δ can be chosen to provide optimum results for the particular noise environment and no external reference noise signal need be provided.

This technique has disadvantages also. First, no subjective experimental results are provided for performance for the filter in conjunction with LPC analysis/synthesis. Research has indicated that echo problems exist with time domain implementations of Wiener filters (Ref 2:694). Lastly, the signal bandwidth must be significantly less than the bandwidth of the additive noise.

Frequency Domain Linear Prediction Filter

This technique attempts to modify the LPC analysis/synthesis process to account for corruption of the speech with noise. Specifically, the speech extraction problem is regarded as a parameter estimation problem (Ref 6).

The method assumes that the power spectral density of the noise is known (noise is stationary) and that the statistics of speech and noise are both gaussian. A periodgram of windowed speech corrupted with noise is calculated by Eq (3).

$$|X_T(f)|^2 = |S_T(f)|^2 + |D_T(f)|^2 + 2 \cdot \mathrm{Re}\,[S_T(f) \cdot D_T^*(f)] \qquad (3)$$

$X_T(f)$, $S_T(f)$ and $D_T(f)$ are the Fourier transforms of windowed noise corrupted speech, speech, and noise, respectively. The unbiased estimate of the speech spectrum is given by Eq (4).

$$|S_T(f)|^2 = |X_T(f)|^2 - E[|D_T(f)|^2 + 2\,\mathrm{Re}\{S_T(f)\;D_T^*(f)\}] \qquad (4)$$

This estimate of the speech spectrum is smoothed with a spectral window producing the spectral envelope. Then inverse Fourier transforming the spectral envelope gives the autocorrelation coefficients used in linear predictive analysis (Ref 13:401-403).

The advantage of this method is that noise reduction can be tailored for the particular noise environment encountered by the system.

The disadvantage of this method is that it is not simple to modify existing LPC analysis/synthesis systems to estimate the spectral envelope in this manner.

Vocoder Filter Bank Analysis

This techniuqe performs a spectral decomposition of noisy speech via channel vocoder analysis and attenuates each spectral component depending on how much the measured speech plus noise power exceeds an estimate of the background noise power (Ref 11). This filter would be used as a preprocessor before LPC analysis/synthesis.

A two state model for the speech event is applied in determining the maximum likelihood estimator of the speech power. This model resulted in a class of suppression curves which permits a tradeoff of noise suppression against speech distortion. Real-time experiments have shown that the noise can be made imperceptible by proper choice of a suppression factor, but distortion increases as the input's signal to noise ratio decreases.

The noise suppression filter consists of a bank of second order Butterworth bandpass filters which span the frequency range 120-3270 Hertz. Figure 4 is a block diagram of this system. Measurements must be made to determine the instantaneous signal power and the average signal power at the output of each of the channel filters in order to compute the channel gains. Experimentation shoed that a four second

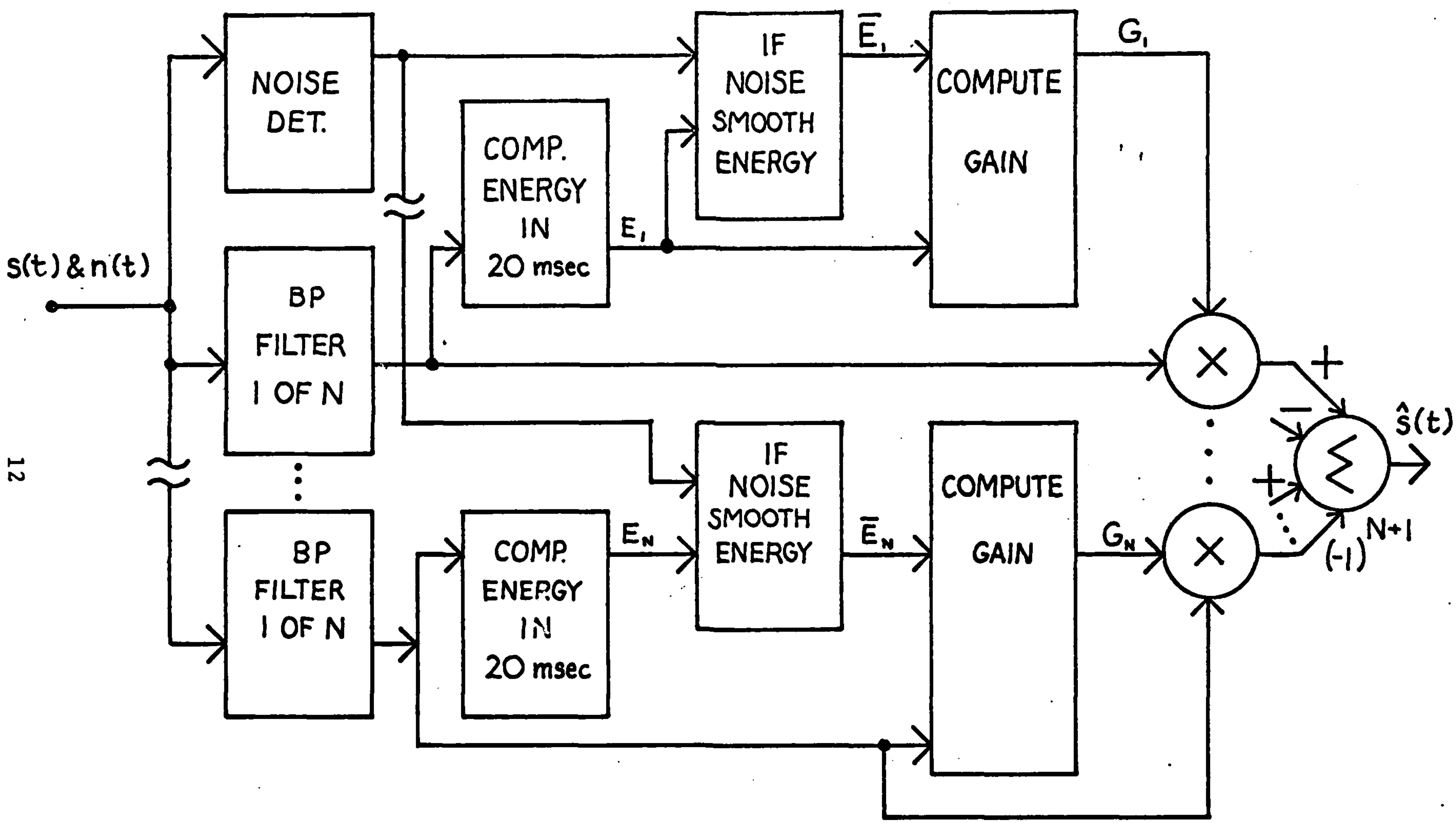


Figure 4. Old Approach to Spectral Subtraction Noise Cancellation

histogram of the frame energies of the signal was bimodal (Ref 11:700). A threshold could be set between the modes, and frames which were speech absent could be determined with high probability. Equation (5) defines the nth channel measurement parameter ($g_n(m)$).

$$g_n(m) = \frac{V_n(m) - V_n(m-1)}{V_n^2(m)} \tag{5}$$

A soft decision point of view determines a class of suppression curves defined by Eq (6).

$$G_n(m) = \frac{1}{2}\left(1 + \sqrt{g_n(m)}\right) \frac{\exp(-\xi) I_o\left(2\sqrt{\xi/(1 - g_n(m))}\right)}{1 + \exp(-\xi) I_o\left(2\sqrt{\xi/(1 - g_n(m))}\right)} \tag{6}$$

$$\xi = \text{suppression factor} \tag{7}$$

$$I_o(x) = \frac{1}{2\pi} \int_0^{2\pi} \exp(x \cos \theta) d\theta \quad \text{(modified Bessel function)} \tag{8}$$

In a real-time implementation, the measurement parameter $g_n(m)$ is used as a pointer for a table look-up to determine the attenuation $G_n(m)$. To avoid discontinuities, a smoothed gain $\overline{G}_n(m)$ is calculated and applied to the appropriate channel. The channel waveforms are then added together to produce the prefiltered waveform.

This prefilter has three primary advantages. First, it is possible to select a suppression factor which would optimize

intelligibility for a given signal to noise ratio. Secondly, it is possible to integrate the prefilter with efficient channel vocoder implementations. Lastly, no reference noise signal must be provided.

The disadvantage of this method is that it is relatively more complex and would be more difficult to implement.

Linear Prediction Coefficient MAP Estimation

The Maximum A Posteriori (MAP) estimation procedure can be used to estimate the LPC coefficients from speech waveforms degraded by additive white gaussian noise. But this procedure requires solving a set of non-linear equations which require too much computation time. However, the true MAP estimation procedure can be approximated by an iterative method that requires the solution of sets of linear equations (Ref 8).

Equation (9) describes noisy speech $y(n)$ as the sum of speech $(s(n))$ and white gaussian noise $(d(n))$.

$$y(n) - s(n) + d(n) \tag{9}$$

The LPC coefficients can be written in vector form $\underline{a}$. The MAP estimate of $\underline{a}$ is the vector that maximizes $p(\underline{a}/\underline{y})$ (the probability density of $\underline{a}$ conditioned on $\underline{y}$). It can be shown that maximizing $p(\underline{a}/\underline{y})$ is a non-linear problem (Ref 8). A "suboptimal" MAP procedure is proposed which estimates $\underline{s}$ and $\underline{a}$ by maximizing $p(\underline{a}, \underline{s}/\underline{y})$. In the iterative procedure, an

initial estimate $\hat{\underline{a}}_o$ is obtained, then $\underline{s}$ is estimated by $E\{s/\hat{a}_o, \underline{y}\}$. With estimate $\underline{s}$, a new estimate $\hat{\underline{a}}_1$ is obtained from linear predictive analysis. With the new $\hat{\underline{a}}_1$, the above procedure is repeated obtaining $\hat{\underline{a}}_2$, etc. Estimating $\underline{s}$ by $E\{\underline{s}/\underline{a}_i, \underline{y}\}$ is a linear problem and this iterative procedure converges to a solution that is at least a local maximum of $p(\underline{a}, \underline{s}/\underline{y})$ (Ref 13:201-203). Also, each estimate of $\underline{s}$ by $E\{s(n)/\hat{\underline{a}}, \underline{y}\}$ can be approximated by filtering y(n) by a non-causal Wiener filter (Eq (10)).

$$H(w) = \frac{P_s(w)}{P_s(w) + P_d(w)} \tag{10}$$

$P_d(w)$ is the power spectral density of the noise.

The primary disadvantage of this technique is that the development is done for the white gaussian noise case. A viable noise tolerant estimation of LPC coefficients must take into account different noise environments.

Phase Corrected Spectral Subtraction

Spectral subtraction noise cancellation preprocessing has been implemented to improve the quality of LPC speech. This method helps, but it introduces "musical noise" problems. A new approach to spectral subtraction noise cancellation has been proposed which avoids the artificial phase distortion which is said to produce the "musical noise" (Ref 10).

Figure 5 is a block diagram of the old approach to spectral subtraction noise cancellation. The time domain output signal is constructed using the phase of the signal plus noise input. Because of this artificial phase distortion, the output does not conform to a linear predictive model. According to the new approach, this problem is eliminated when the subtraction is done at the point where the autocorrelation is calculated or when the covariance matrix is computed in linear prediction analysis.

Figure 6 is a diagram of the new approach in which the autocorrelation lags are modified. If the input noise is assumed white, then the pre-whitening filter can be discarded.

The covariance matrix can be modified also to effect noise cancellation. For white noise, only the diagonal terms of the covariance matrix need be reduced by the noise power for the matrix to conform to the noise-free case. The non-white noise case would require a time varying pre-whitening filter.

The primary advantage of this method is that no noise reference channel need be provided.

Noise Cancellation Using the Short Time Transform

The time domain implementation of a Wiener least squares filter to preform noise cancellation is ineffective when the noise characteristics, e.g., mean, variance, etc. change

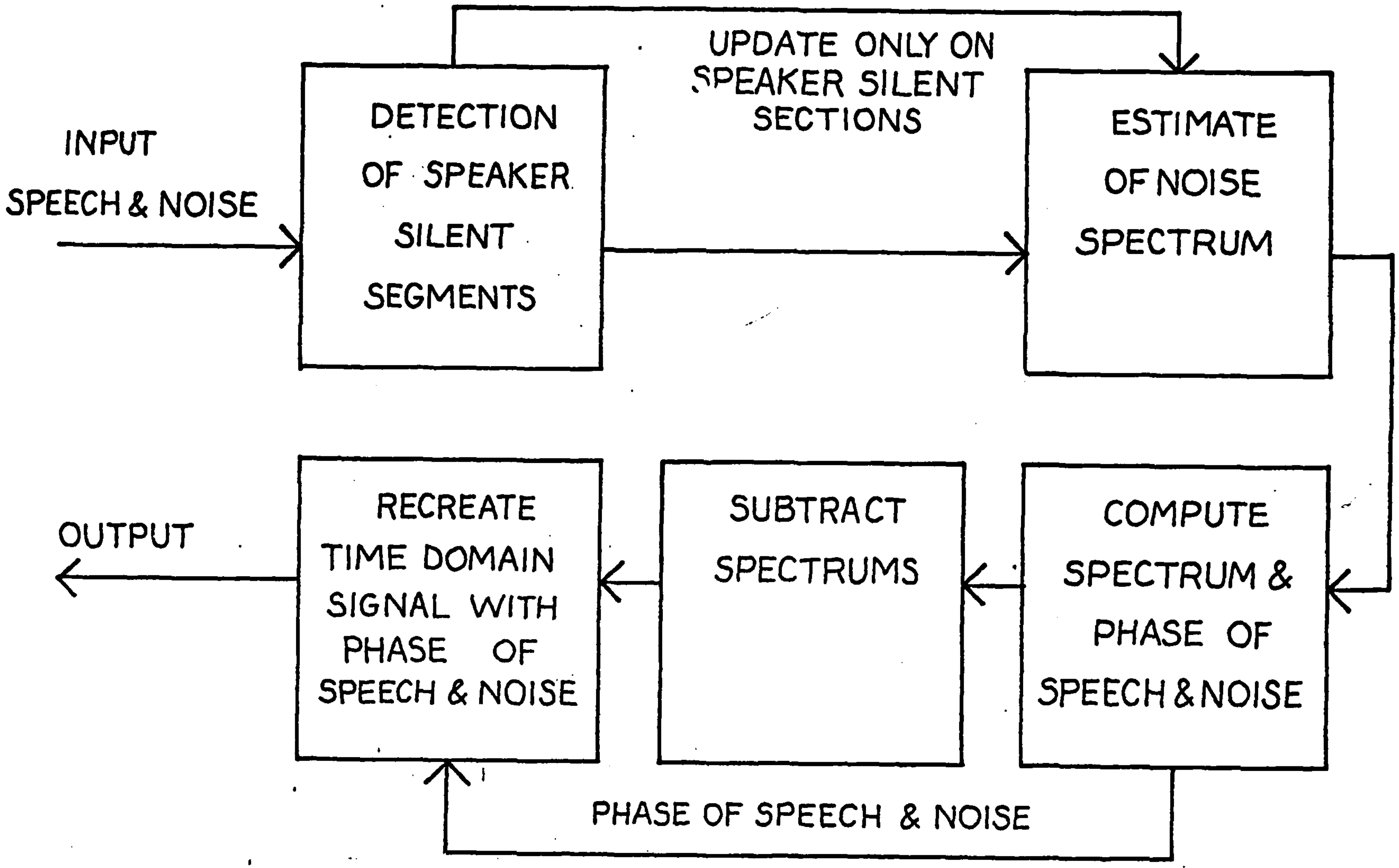


Figure 5. Filter Bank Noise Suppression

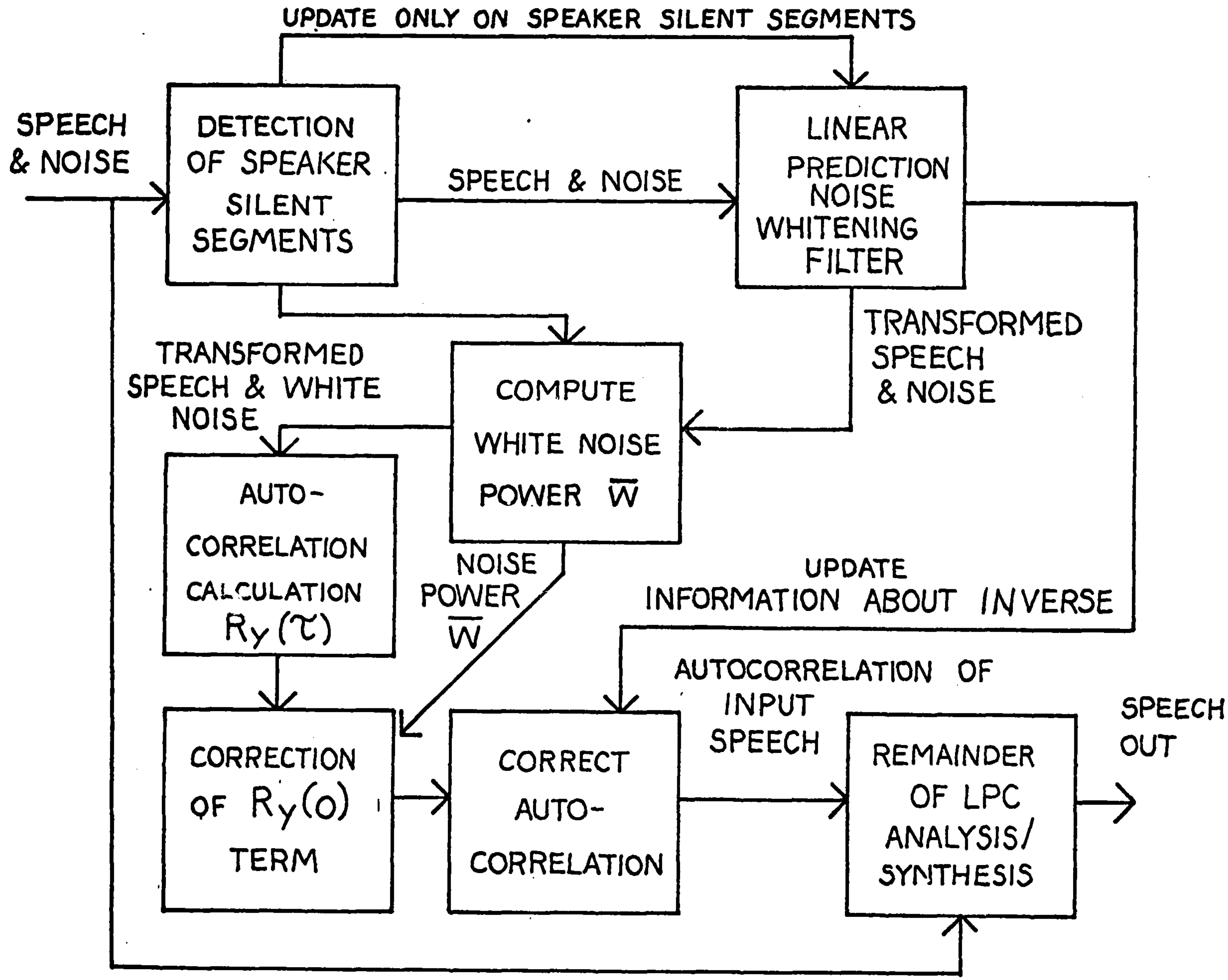


Figure 6. New Approach

rapidly with time. Thus, a frequency domain approach using the short time Fourier transform (FFT) to estimate the required Wiener filter is proposed (Ref 2). Using the efficiency of the FFT results in a computation rate which is proportional to the filter length times the log of the filter length. Therefore, the FFT approach is a viable alternative for real time implementation.

Figure 7 is a block diagram of the filter's construction. The signal x represents speech plus noise and v represents a noise reference channel. $S_{vx}(f)$ is the cross power spectral density between speech plus noise and noise. $S_{vv}(f)$ is the power spectral density of the noise. W(f) is the estimated Wiener filter. A more complete description of this filter will be given in the next section, because this filter was chosen for implementation.

There are three primary advantages of this approach. First, experimental comparisons have shown that the computational efficiency is 3.5 times as great as time domain methods. This means that, for reverberant high noise environments requiring large filter lengths, implementation in real time may be accomplished. Secondly, although the filter structure shown in Figure 7 requires a reference noise channel, it is possible to obtain estimates of the background noise spectrum during speaker-silent segments. Lastly, no a priori information about the energy in the reference channel is needed in order to pick a prediction

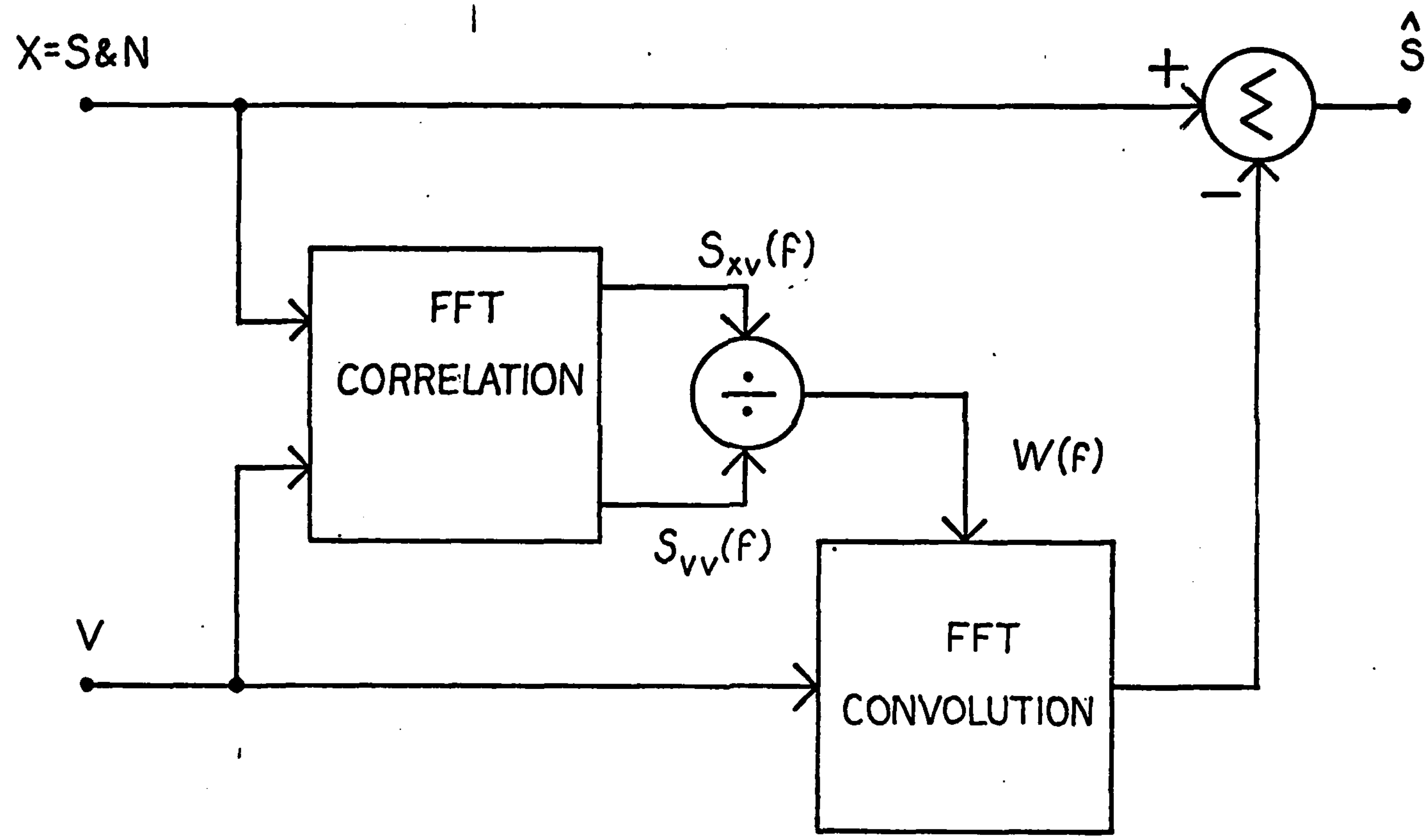


Figure 7. Block Diagram of Short Time Transform Noise Cancellor

distance as in some time domain methods. If the prediction distance is marginally too large, then the algorithm will produce echo. If too small, the algorithm will be slow to converge. This gain adjustment is taken care of automatically in the frequency domain approach. Therefore, higher quality output is produced free from echo which has been demonstrated already.

There are two disadvantages of this method. It will require considerably more memory than time domain methods. Also, it will be much more complex to program than time domain methods.

III. Implementation of Noise Cancellation Technique

I chose to implement the short-time transform technique of noise cancellation. This technique was described in the previous section and is depicted in Figure 8. This technique was chosen for the following reasons. First, it is a preprocessor and thus can be used directly to improve LPC processing without modification of the LPC implementation. Also, as a preprocessor, it may be used to enhance the performance of other speech processing procedures being developed (one example is phoneme recognition). Secondly, this frequency domain technique had faster adaptation time than time domain implementations of the Wiener filter (Ref 2). Thirdly, preliminary subjectivity results indicated that frequency domain methods do not have echo problems, as discussed in the last section. Lastly, it could be implemented in the time allowed with greater ease than the frequency domain vocoder method discussed earlier. The frequency domain vocoder shares with the short-time transform method the advantages of faster adaptation time and lack of echo problems, but is much more complex and more difficult to implement. Therefore, the short-time transform method was chosen over methods which modified the LPC implementation, were implemented in the time domain, and required too much complexity for implementation in the time allotted.

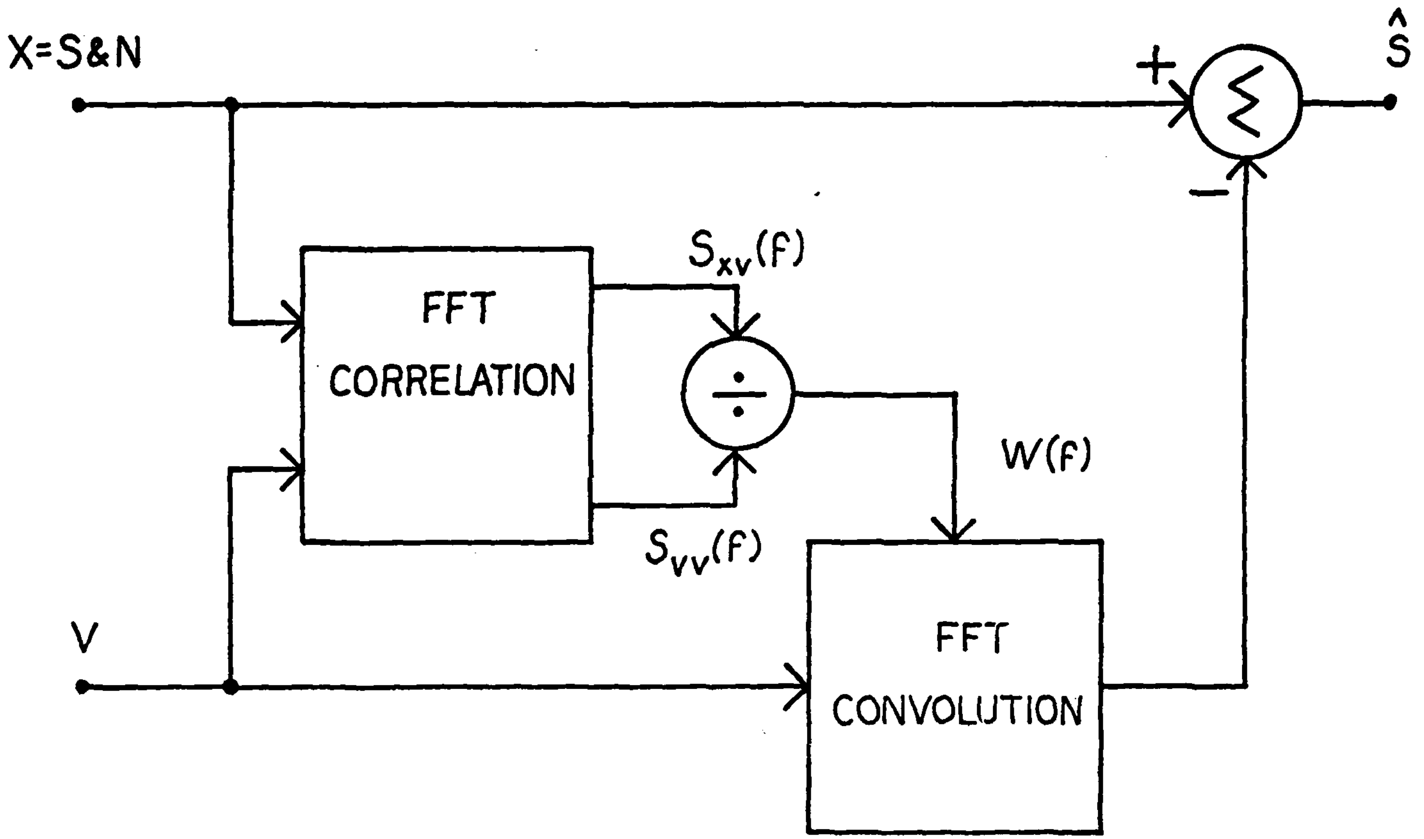


Figure 8. Block Diagram of Short-Time Transform Noise Cancellor

General System Information

This technique was implemented in Fortran 5 on a Data General Nova-Eclipse signal processing system. Analog speech data are low-pass filtered at 4.0 kilohertz and sampled at 8000 samples/second (approximately the Nyquist rate). Speech files consist of 88 blocks (256 words/block) of integers ranging from -2048 to 2048 (-5 to +5 Volts). More specific information about input/output of speech files through program AUDIOHIST, written by Paul Finkes, is available (Ref 14).

Short-Time Transform Noise Cancellation

The implementation of this frequency domain method is based on an article by Lawrence R. Rabiner and Jant B. Allen, titled "Short-Time Fourier Analysis Techniques for FIR System Identification and Power Spectrum Estimation" (Ref 14). Specifically, Rabiner and Allen define the short-time Fourier transform of a signal x(n), at time mR as Eq (12):

$$\mathbf{X}(m,k) = \sum_{x=mR-L+1}^{mR} x(n)w(mR-n)\exp(-j\,\frac{2}{N}\,kn) \tag{12}$$

R is the period between estimates of the short-time transform of the signal and w(n) is a causal FIR window of duration L samples. They show if the signal plus noise signal input is defined as x(n) and the noise input v(n), then the unbiased Wiener filter estimate is given by Eq (13).

$$H(K) = \frac{S_{vx}(K)}{S_{vv}(K)} \tag{13}$$

where

$$S_{vx}(K) = \sum_{m=0}^{p-1} \sum_{q=q_1}^{q_2} X(m,K)\ V^*(m+q,K) \tag{14}$$

$$S_{vv}(K) = \sum_{m=0}^{p-1} \sum_{q=q_1}^{q_2} V(m,K)\ V^*(m+q,K) \tag{15}$$

and

q_1 = integer part of $[(L+\hat{M}-2)/R]$

q_2 = integer part of $[(L-1)/R]$

$\hat{M}$ = estimate of the system's impulse response duration

p = number of analysis sections.

Bounds are defined for the parameters L and R in Eqs (16) and (17).

$$L \geq \hat{M} \tag{16}$$

For a Hamming window,

$$R \leq L/4 \tag{17}$$

For this implementation, $\hat{M}$ was chosen to be equal to L or 128 samples (16 msec). R is chosen to be 32 samples (4 msec). The parameters q_1 and q_2 are then calculated as -7 and 3. The number of analysis sections p is left as a variable whose effect is to be evaluated because no information relating to this specific application was available in order to choose a value. Rabiner and Allen do show that, as p increases, less error is made in the estimate H(K) (Ref 14: 190-191). The parameter p is related to the total number of points used in the analysis N′ by Eq (18)

$$p \;=\; \text{integer part}\;\left[\frac{(N'-L+R)}{R}\right] \tag{18}$$

The process of noise cancellation is implemented can be described as follows. First, weight 128 point sequences of both channels with Hamming windows and compute N point FFT by zero filling from 129 to N. Repeat previous step (except shift 32 samples) until $p + q_2 - q_1$ FFTs have been calculated and stored. Then, calculate $S_{vx}(K)$ and $S_{vv}(K)$ according to Eqs (14) and (15) by correcting the relative phase between successive FFTs at each analysis frame M due to the shift of 32 samples between successive FFTs. H(K) is then calculated according to Eq (13). A FFT of the noise reference channel V(0,K) is multiplied by H(K) and then inverse Fourier transformed to produce the sampled sequence of correction signal

on the interval 1 to 128. The stored FFTs of the following two channels are updated by shifting in memory and storing one new FFT of both channels in the space opened up after shifting. Then H(K) is calculated again as above, repeating the process. A new sampled sequence of the correction signal is produced and added with a 75% overlap of the previous sequence calculated. After reconstruction, the correction sequence is subtracted from the speech plus noise channel, completing the operation. Appendix A gives a description of how this process was implemented.

IV. Results

The short-time noise cancellation implementation was used to process files consisting of 2.0 seconds of speech plus noise. Each speech plus noise file was generated with a predefined speech to noise ratio by program SPLUSN. The noise used as the reference channel and by SPLUSN is white gaussian and generated by program NOISE. Both NOISE and SPLUSN are described in Appendix A.

The time required to process each 2.0 second speech file using six analysis frames per estimation was excessive (approximately 5.2 hours). The execution time is approximately proportional to the number of frames per estimation since processing with three analysis frames per estimation required 2.6 hours. The total run-time can be approximately broken down into three areas. It took 27.8% of the time to read and write complex arrays to memory. It took 4.22% of the time to calculate 1024 point DFTs. Also, it took 72.16% of the time to perform all other processing which mostly included complex arithmetic and logical operations.

The noise cancellor performance is depicted by Figures 9, 10 and 11. Figure 9 is a plot of four sequential segments (each segment is .128 seconds long) of speech. Figure 10 is a plot of the same speech segments plus noise (SNR = 0 db, as defined by program SPLUSN in Appendix A using B weighting curves on speech and noise energies). Figure 11 is a plot of the output of the noise cancellor when the speech plus noise

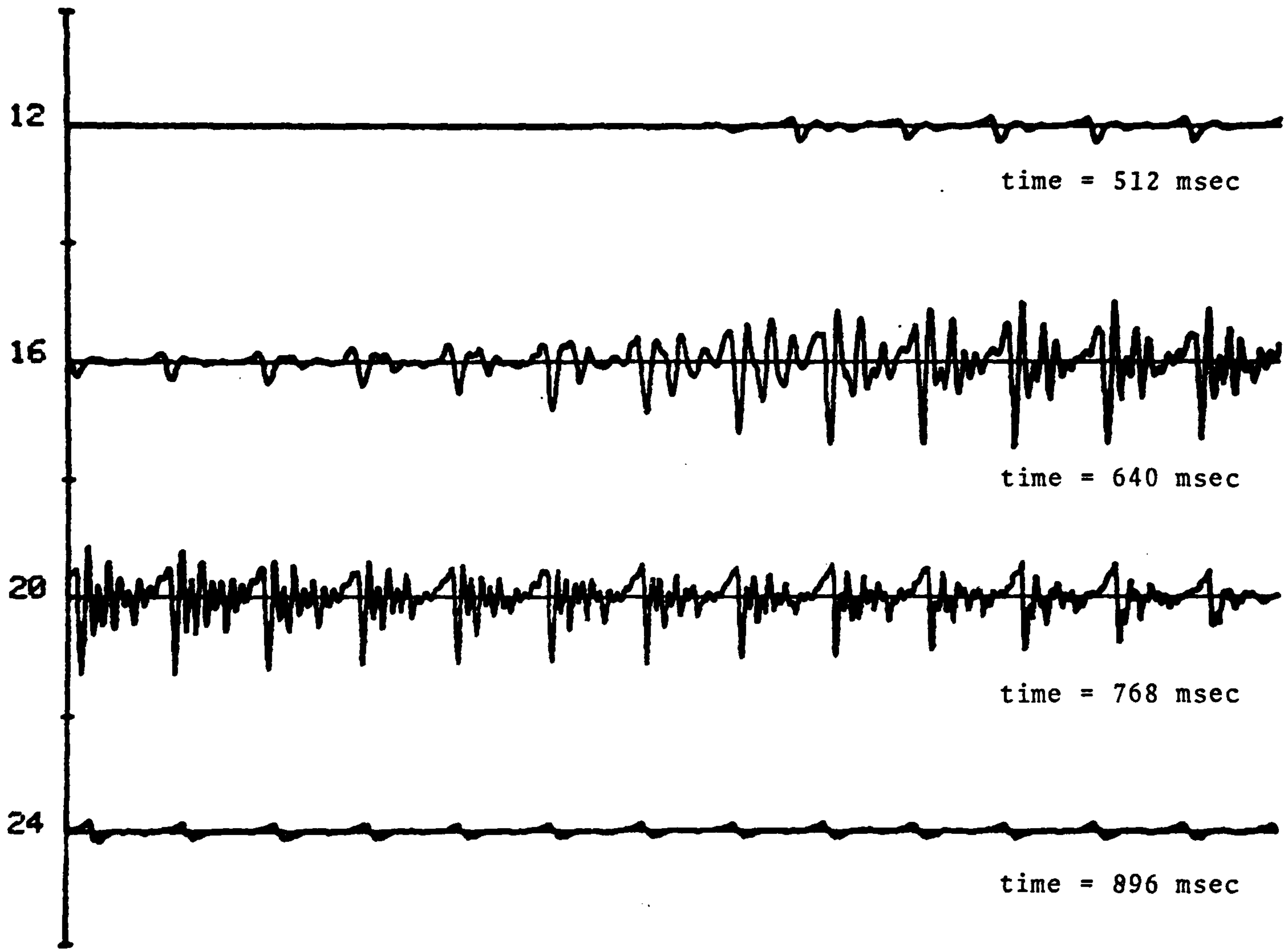


Figure 9.   Speech Waveform

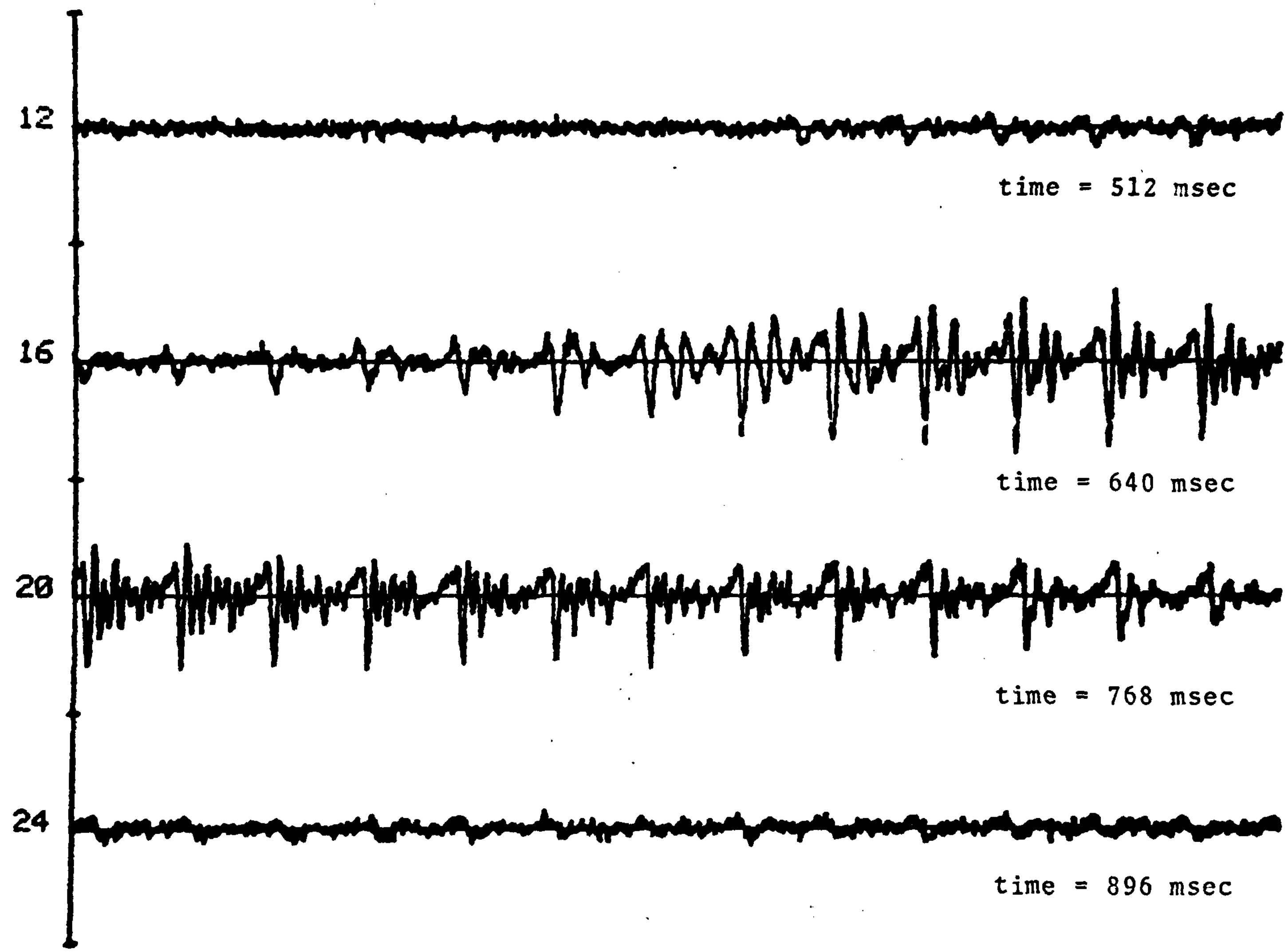


Figure 10. Speech Plus Noise Waveform

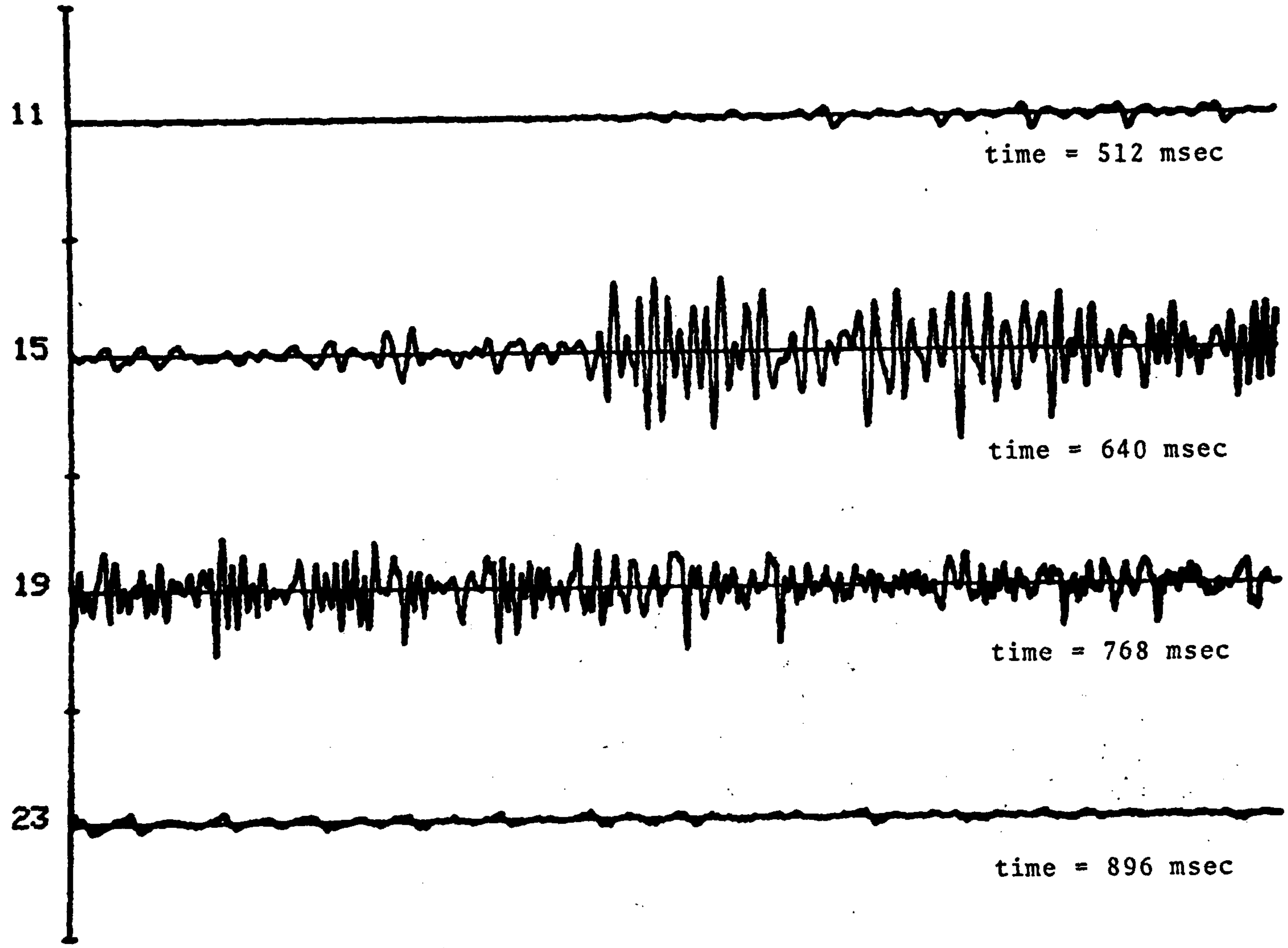


Figure 11. Noise Cancelled Speech

in Figure 10 is used as its input and six analysis frames were used per estimation. Note that little noise appears in the output, but the speech waveform is distorted. It is more difficult to visually pick out the glottal pulse in the speech waveform. I listened to output speech and it sounded like whispered speech. Also, little noise was heard in the output.

Noise cancellation was performed with three and six analysis frames per estimation.

It was noted that the output speech was less whisper-like (higher quality) with six analysis frames per estimation. However, the output speech was still whisper-like and of low quality.

Next, I used our Interactive Laboratory System (ILS) to perform linear predictive analysis/synthesis on speech plus noise and noise cancelled speech. The use of the ILS system in this application is described in Appendix B. Also, the details on how the ILS system performs linear predictive analysis/synthesis are given in the ILS application note number 1 entitled "Speech Analysis and Synthesis" (Ref 15).

The number of points per analysis frame used in the LPC analysis was chosen to be 128. The number of coefficients estimated was chosen to be 10. Also, analysis frames were windowed with a standard Hamming window.

The synthesized speech is depicted in Figures 12 and 13. Figure 12 is the LPC processed version of the speech plus noise shown in Figure 10. Figure 13 is the LPC processed

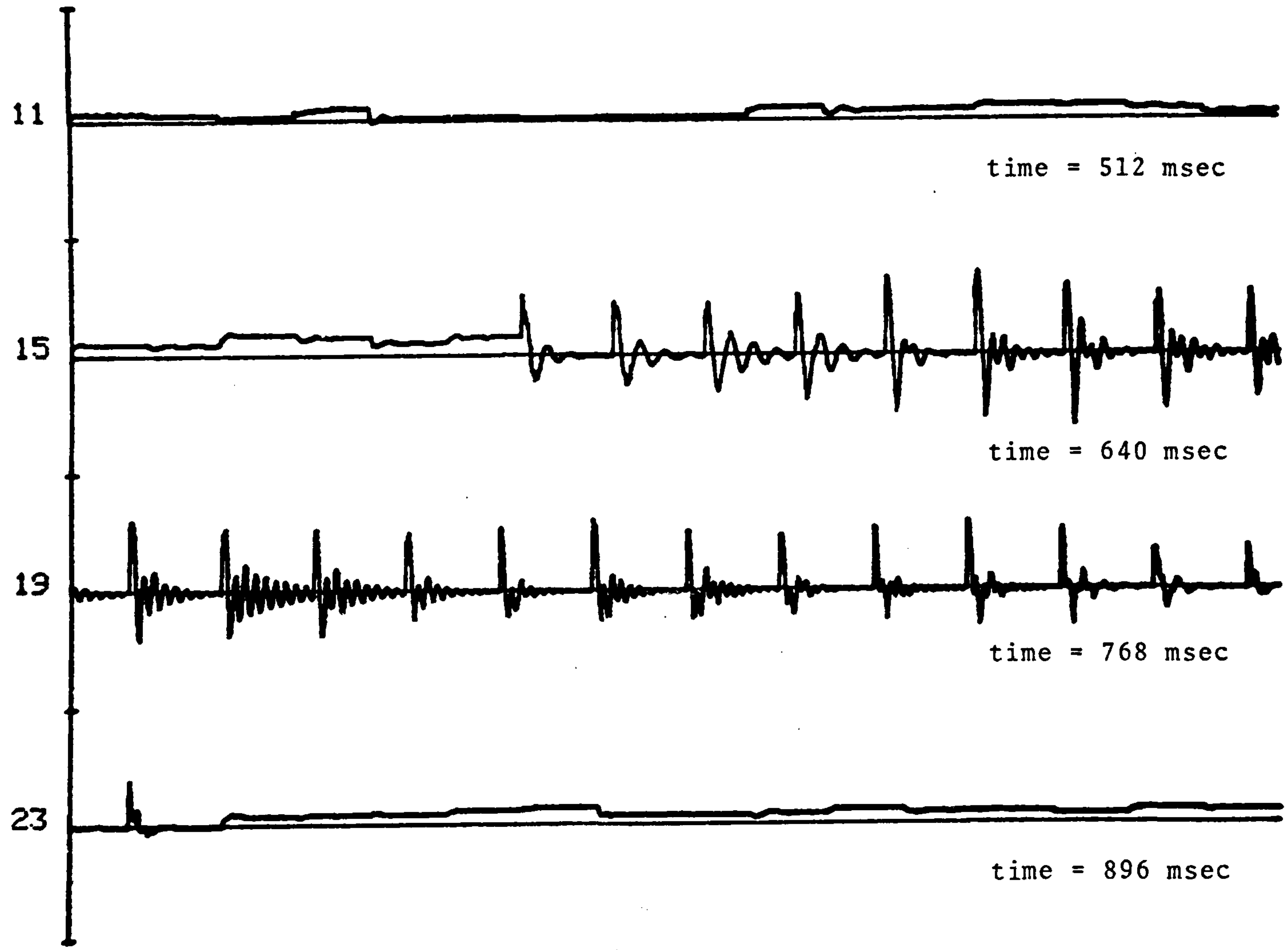


Figure 12. LPC Synthesized Speech Plus Noise

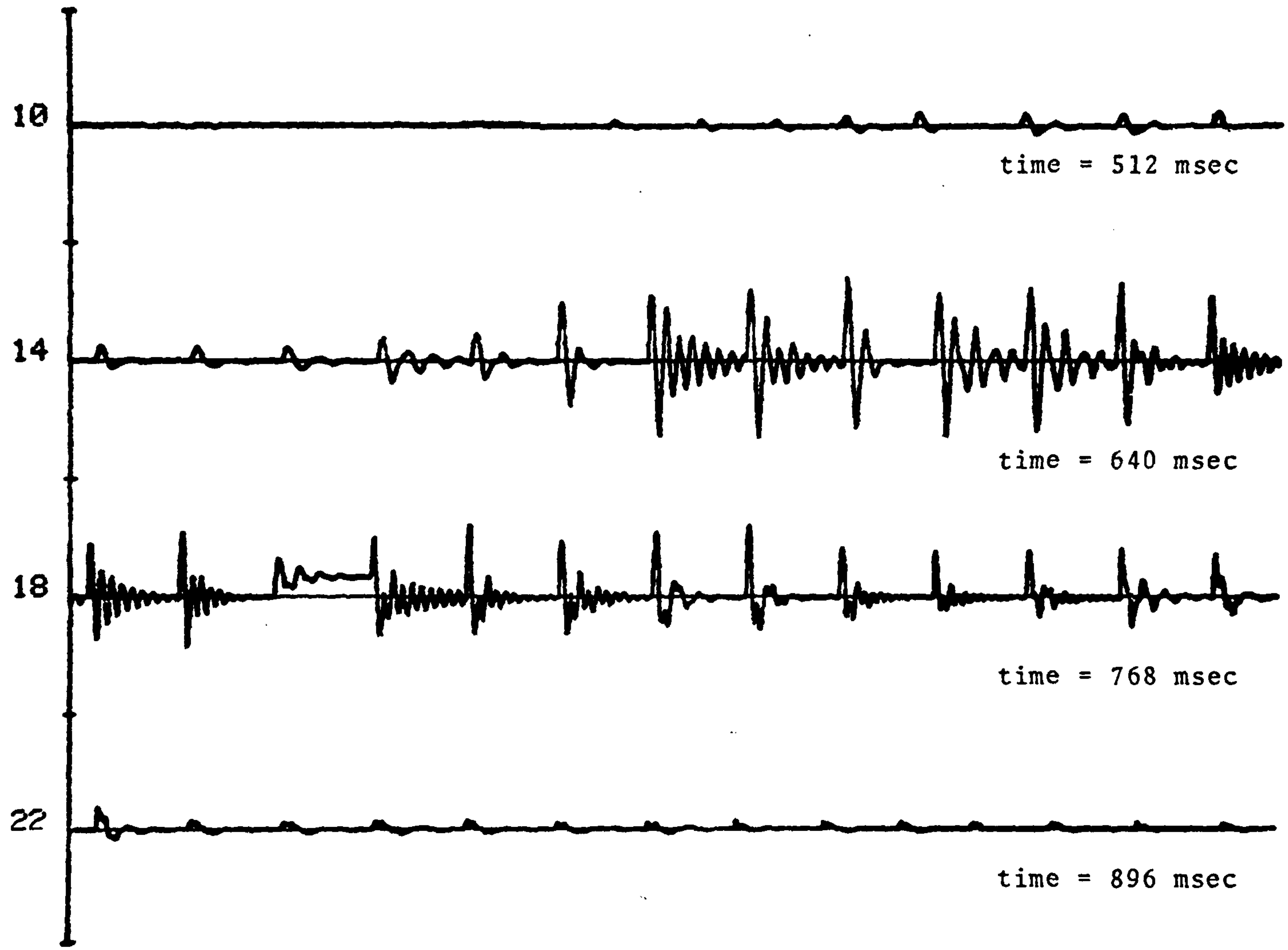


Figure 13. LPC Synthesized Noise Cancelled Speech

version of the noise cancelled speech shown in Figure 11. After listening, the LPC processed speech plus noise was perceived to be of very poor quality. The LPC processed noise cancelled speech retained some of its whisper-like quality, but was of much higher quality than the LPC processed speech plus noise.

V. Conclusions

This short-time noise cancellation method was capable of improving the quality of LPC processed speech plus noise, but this implementation has two major weaknesses. First, this implementation takes an excessive amount of time to perform noise cancellation. Secondly, the output speech produced by this implementation sounded like whispered speech.

This implementation takes an excessive amount of time for the following three reasons. First, 27.8% of the processing time is devoted to reading and writing 1024 point complex arrays (4096 words) to disk. Secondly, the Data General Nova/Eclipse performs floating point instead of integer complex arithmetic. Integer arithmetic is carried out quicker, but floating point arithmetic allows greater dynamic range in processing. Lastly, the DFT size was chosen to be 1024 and may be reduced to 512 without reducing the performance of the cancellor. A DFT size of 512 would cut in half the time necessary to read and write complex arrays to disk and the time required for complex arithmetic operations.

The whispered speech effect of the output from the cancellor was reduced when the number of analysis frames used per estimation was doubled from three to six. The effect of increasing the analysis frames used in each calculation of estimates of the cross-spectrum and the auto-

spectrum is that these spectra are smoothed more over time. Thus, the Wiener filter estimate does not change as rapidly over time and will not produce as much modulation of the speech waveform. Additional smoothing could be implemented by smoothing the spectrum estimates by taking partial sums of past and present spectrum estimates.

VI. Recommendations

The areas for future research into the characteristics of performance of this implementation of the short-time transform noise cancellation include the following:

1. Whether or not the current implementation is modified, formal subjective testing (diagnostic rhyme test) should be carried out with LPC processed speech plus noise and LPC processed noise cancelled speech. This testing would provide a more firm basis for describing the performance of the noise cancellor.

2. The addition of an array processor on the Data General Eclipse computer could greatly increase the speed of complex array processing. This noise cancellor would have to be modified to allow the array processor to perform arithmetic operations on complex arrays.

3. The addition of more memory allocated for programs may make it feasible to store FFTs in extended memory instead of on disk. Thus, the program would not have to perform input/output operations to disk which take 27.8% of the current processing time.

4. This noise cancellor's FFT size of 1024 points could be halved to 512 points and cut in half the processing time. But, it must be determined whether

or not the decreased resolution in frequency reduces the noise cancellor's ability to reject single interfering tones.

5. Spectral smoothing of the power spectrum estimates would probably improve the quality of the noise cancelled speech. This could be accomplished by taking partial sums of past and present power spectrum estimates.

6. The current implementation requires an external noise reference channel. This implementation could be modified to update the estimate of the power spectrum of the noise on speaker silent segments from the speech *plus noise channel. This* requires detection of the speaker silent segments by thresholding. The threshold is chosen by studying four second histograms of speech plus noise which are bimodal. A threshold is picked from between the two modes.

Bibliography

1. S.T. Alexander, E.H. Satorius, and J.R. Zeidler, "Linear Predictive Digital Filtering of Narrowband Processes in Additive Broadband Noise," NOSC TR 331, Naval Ocean Systems Center, San Diego CA 92152.
2. Steven F. Boll, "Adaptive Noise Cancelling in Speech Using the Short-Time Transform," Proceedings 1980 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 692-695, April 9-11, 1980, Denver CO.
3. Steven F. Boll, "Noise Suppression Methods for Robust Speech Processing," Semi-Annual Technical Report to Defense Advanced Research Projects Agency, Contract Number N00173-77-C-0041, April 1977.
4. W.J. Done, "Estimating the Parameters of a Noisy All-Pole Process Using Pole-Zero Modeling," Proceedings 1979 IEEE Conference on Acoustics, Speech and Signal Processing, pp. 228-231, 2-4 April 1979, Washington DC.
5. Paul Finkes, "G Stress Effects on Speech Recognition," MS Thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, December 1981.
6. Hidefumi Kobatake, Junta Inari and Shin-ichi Kakuta, "Linear Predictive Coding of Speech Signals in a High Ambient Noise Environment," Proceedings 1978 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 472-475, April 10-12, 1978, Tulsa OK.

7. B. Gold, "Robust Speech Processing," Technical Note 1976-C, Lincoln Laboratory, M.I.T., January 27, 1976, DDC ADC-AD-A012P9910.

8. Jae S. Lim, "Estimation of LPC Coefficients from Speech Waveforms Degraded by Additive Random Noise," Proceedings of the 1978 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 599-601, 10-12 April 1978, Tulsa OK.

9. Jae S. Lim, Alan V. Oppenheim, "All-Pole Modeling of Degraded Speech," IEEE Transactions on Acoustics, Speech, and Signal Processing, pp. 197-209, June 1978.

10. Sidhartha Maitra, "Reducing the Effect of Background Noise for Low-Bit-Rate Voice Digitizers," Proceedings 1980 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 696-698, April 9-11, 1980, Denver CO.

11. Marilyn L. Malpass, Robert J. McAulay, "A Real-Time Noise Suppression Filter for Speech Enhancement and Robust Channel Vocoding," Proceedings 1980 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 699-702, April 9-11, 1980, Denver CO.

12. L.R. Rabiner and J.B. Allen, "Short-Time Fourier Analysis Techniques for FIR System Identification and Power Spectrum Estimation," IEEE Transactions on Acoustics, Speech, and Signal Processing, pp. 304-314, August 1978.

13. L.R. Rabiner and R.W. Schafer, Digital Processing of Speech Signals, Prentice Hall, Englewood Cliffs NJ, 1979.

14. M.R. Sambur and N.S. Jayant, "LPC Analysis/Synthesis from Speech Containing Quantizing Noise or Additive White Noise," IEEE Transactions on Acoustics, Speech and Signal Processing, pp. 488-494, December 1976.

15. Signal Technology, Inc. "Application Note No. 1, Speech Analysis and Synthesis," Manual for Interactive Laboratory System. Signal Technology, Inc., 15 W. De La Guerra Street, Santa Barbara CA, March 1981.

APPENDIX A

Software

Noise Cancellation Process

The noise cancellation process consists of three programs executed in sequence. First, BEGNCL obtains names of I/O files and number of analysis frames per estimation from user and stores them for the following programs to read. Second, NCANCEL performs the calculations necessary to produce the (correction signal) time domain estimate of the noise. Thirdly, SUBTRACT subtracts the time domain estimate of the noise from the speech plus noise file. This entire process is executed by running macrofile NC.MC. Figure 14 is a flowchart of the overall process. The following source listings explain each program.

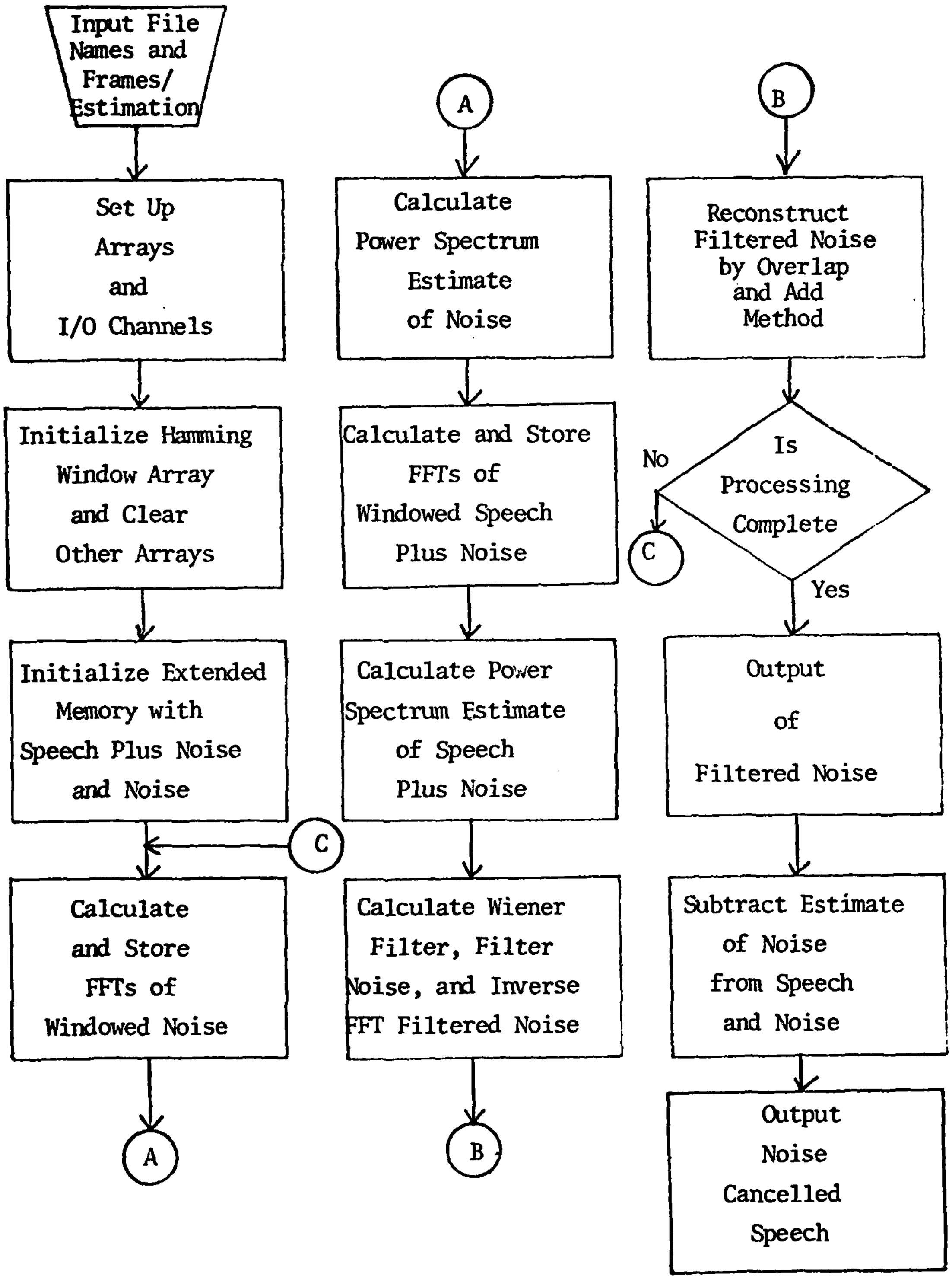


Figure 14. Flowchart of Implementation of Noise Cancellor

```
C       ************************************************************
C       ************************************************************
C       **                                                        **
C       **      PROGRAM:   BEGNCL                                 **
C       **                                                        **
C       **              AUTHOR: LT CHRIS BATCHELOR                **
C       **                                                        **
C       **                      DATE:   DEC. 1, 1981              **
C       **                                                        **
C       ************************************************************
C       ************************************************************
C
C       THIS PROGRAM ACCEPTS NAMES OF FILES USED IN NOISE CANCELLATION
C       AND THE NUMBER OF ANALYSIS FRAMES PER ESTIMATION OF POWER
C       SPECTRUMS.  THESE FILENAMES ARE STORED IN DISK FILE "NAME"
C       ALONG WITH THE NUMBER OF ANALYSIS FRAMES PER ESTIMATION.
C
        DIMENSION NSPFIL(7),NOISFIL(7),NOUTFIL(7)
        TYPE" THIS PROGRAM PERFORMS NOISE CANCELLATION"
        ACCEPT" SPEECH PLUS NOISE FILENAME? "
        READ(11,1)NSPFIL(1)  ;read filename of speech plus noise
        ACCEPT" NOISE FILENAME? "
        READ(11,1) NOISFIL(1)  ;read filename of noise
        ACCEPT" OUTPUT FILENAME? "
        READ(11,1) NOUTFIL(1)  ;read filename for output
1       FORMAT(S13)
        ACCEPT" INPUT ANALYSIS FRAMES/ESTIMATION ",NANAL
        OPEN 2,"NAME",ATT="B",REC=1  ;open file containing filenames and NANAL
        WRITE(2,1)NSPFIL(1)  ;store speech plus noise filename
        WRITE(2,1)NOISFIL(1)  ;store noise filename
        WRITE(2,1)NOUTFIL(1)  ;store output filename
        WRITE(2,2)NANAL  ;store # of analysis frames/estimation
2       FORMAT(I4)
        CALL RESET  ;close channel
        END
```

```
C          ********************************************************
C          ********************************************************
C          **                                                    **
C          **      PROGRAM:   NCANCEL                            **
C          **                                                    **
C          **               AUTHOR: LT CHRIS BATCHELOR           **
C          **                                                    **
C          **                       DATE:   NOV 2, 1981          **
C          **                                                    **
C          ********************************************************
C          ********************************************************
C
C              THIS PROGRAM CALCULATES A WIENER FILTERED CORRECTION
C          SIGNAL WHICH IS SUBTRACTED (IN SUBTRACT) FROM NOISE CORRUPTED
C          SPEECH TO ACCOMPLISH NOISE CANCELLATION.  THE WIENER FILTER
C          IS CALCULATED FROM POWER SPECTRUM ESTIMATES USING UNBIASED
C          SHORT-TIME FOURIER ANALYSIS.  EACH POWER SPECTRUM ESTIMATE
C          IS CALCULATED FROM PAST AND FUTURE FOURIER TRANSFORM
C          PRODUCTS, THEREFORE ELIMINATING THE BIAS IN POWER SPECTRUM
C          ESTIMATION DUE TO FINITE WINDOW LENGTH.  THE WIENER FILTER
C          IS EQUAL TO THE CROSS-SPECTRUM BETWEEN NOISE CORRUPTED
C          SPEECH AND A NOISE REFERENCE CHANNEL DIVIDED BY THE AUTO-
C          SPECTRUM OF THE NOISE REFERENCE CHANNEL.  THE
C          CROSS-SPECTRUM IS CALCULATED BY SUBROUTINE SPCH.
C          FILTERING OF THE NOISE REFERENCE CHANNEL IS DONE BY
C          MULTIPLYING BY THE WIENER FILTER ESTIMATE. RECONSTRUCTION
C          OF THE FILTERED NOISE (CORRECTION SIGNAL) IS ACCOMPLISHED
C          USING THE OVERLAP AND ADD METHOD.
C
C*************************************************************************
C                         VARIABLES AND ARRAYS
C
C          NUMA :  INTEGER ARRAY FOR INPUT AND OUTPUT FOR WINDOWED
C                  TIME SEQUENCES FROM EXTENDED MEMORY
C
C          FOURN : COMPLEX ARRAY FOR USE IN CALCULATING FOURIER TRANSFORMS
```

```
C
C       FOURS :  COMPLEX ARRAY FOR USE IN CALCULATING FOURIER TRANFORMS
C
C       SPEC :   COMPLEX ARRAY CONTAINING POWER SPECTRUM ESTIMATES OF
C                NOISE
C
C       IO1 :    ARRAY IN LABELED COMMON WHICH REPRESENTS THE WINDOW
C                WHICH MAY BE MOVED THROUGH EXTENDED MEMORY BY
C                WINDOW MAPPING, BUT IN THIS PROGRAM REMAINS
C                STATIONARY AT EXTENDED MEMORY BLOCK 0.   THIS
C                BLOCK IS NOT MANIPULATED BECAUSE VFETCH AND VSTASH
C                COMMANDS USE BLOCK 0 OF EXTENDED MEMORY AS A
C                SCRATCH FILE
C
C       IO2 :    ARRAY WHICH IS EQUIVALENCED TO FOURN AND IS USED
C                IN I/O OF FOURIER TRANSFORMS
C
C       IO4 :    ARRAY WHICH IS EQUIVALENCED TO FOURS AND IS USED
C                IN I/O OF FOURIER TRANSFORMS
C
C       HAMMING : COMPLEX ARRAY CONTAINING HAMMING WINDOW WEIGHTS
C
C       IST :    INDEX REPRESENTING EXTENDED MEMORY ELEMENT NUMBER
C                OF THE REFERENCE NOISE CHANNEL.   EACH ELEMENT
C                IS 32 WORDS
C
C       IST2 :   INDEX REPRESENTING EXTENDED MEMORY ELEMENT NUMBER
C                OF NOISE CORRUPTED SPEECH
C
C       ICONT :  INDEX REPRESENTING EXTENDED MEMORY ELEMENT NUMBER OF
C                FILTERED CORRECTION SIGNAL
C
C       NBLK :   THE BLOCK NUMBER REPRESENTING STORAGE LOCATION OF
C                FOURIER TRANSFORMS
C
C       MBLK :   SAME AS NBLK
```

```
C
C       NANAL :  NUMBER OF ANALYSIS FRAMES PER ESTIMATION OF
C                POWER SPECTRUMS
C
C       NUMTF :  NUMBER OF FOURIER TRANSFORMS OF NOISE USED IN CALCULATION
C                OF POWER SPECTRUM
C
C       IE :     ERROR CONTROL FLAG
C
C       IC :     NUMBER OF AVAILABLE 1024 WORD EXTENDED MEMORY BLOCKS
C
C***********************************************************************
C
C                INPUT/OUTPUT FILES
C
C       CHANNEL
C
C       1        CONTAINS STORED FOURIER TRANSFORMS AND SUMS OF
C                FOURIER TRANSFORMS
C
C       2        NOISE REFERENCE CHANNEL
C
C       3        CONTAINS NAMES OF I/O FILES
C
C       4        NOISE CORRUPTED SPEECH
C
C       5        FILTERED CORRECTION SIGNAL
C
C***********************************************************************
C
C**** BEGIN PROGRAM, SET UP ARRAYS AND I/O CHANNELS
C
        DIMENSION NUMA(160),NOISFIL(7),NSPFIL(7),NOUTFIL(7)
        COMPLEX FOURN(1024),SPEC(1024),FOURS(1024),HAMMING(128)
        COMMON /BLKO/IO1(1024)/BLK/IO2(4096),IO4(4096)
        EQUIVALENCE(FOURS(1),IO4(1))
```

```
      EQUIVALENCE(FOURN(1),IO2(1))
      OPEN 5,"NSFILT",ATT="BC",REC=62  ;open correction signal file
      OPEN 3,"NAME"  ;open file containing names of I/O files
      READ(3,100)NSPFIL(1)  ;get name of speech plus noise file
      READ(3,100)NOISFIL(1)  ;get name of noise file
      READ(3,100)NOUTFIL(1)  ;get name of output file
      READ(3,101)NANAL  ;read # of analysis frames/estimation
 100  FORMAT(S13)
 101  FORMAT(I4)
      OPEN 2,NOISFIL  ;open noise file
      OPEN 4,NSPFIL  ;open speech plus noise file
      NUMTF=10+NANAL  ;calculate # of FFTs stored for calculation
      TYPE" NUMBER OF ANALYSIS FRAMES/ESTIMATION= ",NANAL
      OPEN 1,"PSPEC",ATT="BC",REC=((3*NANAL+12)*16) ;open FFT storage file
C
C**** INITIALIZE ARRAYS
C
      DO 1 I=1,128  ;initialize Hamming window array
 1      HAMMING(I)=CMPLX(.54-.46*COS(2*3.14159*(I-1)/127.0),0.0)
      DO 2 I=1,1024  ;clear the following arrays
        SPEC(I)=CMPLX(0.0,0.0)
        FOURN(I)=SPEC(I)
        FOURS(I)=SPEC(I)
        IF(I.LE.160)NUMA(I)=0
 2      CONTINUE
C
C**** SET UP EXTENDED MEMORY WITH INPUT DATA
C
      CALL VMEM(IC,IE)  ;how much ext. memory space is available?
      CALL CHECK(IE)
      IF(IC.LE.23)TYPE"EXT. MEMORY TOO SMALL, BLOCKS= ",IC*4
      CALL MAPDF(IC,IO1,1,32,IE)  ;map ext. memory with elements of 32
C                                  words long and place the window
C                                  array IO1 at block 0
      CALL CHECK(IE)
      CALL ERDB(2,0,4,62,IC,IE)  ;read into ext. mem. noise data
```

```
      CALL ERDB(4,0,66,62,IC,IE)  ;read into ext. mem. speech plus noise
C
C**** START PROCESS LOOP
C
      IST=0
      IST2=0
      ICONT=1025
C
C**** CALCULATE AND STORE FFTS OF WINDOWED NOISE
C
 3    IST=IST+1  ;increment counter
      IF(IST.EQ.494) GO TO 16  ;check for processing complete
      NST=IST+32  ;offset of 32 to first block of ext. mem.
      CALL VFETCH(NUMA,NST,4)  ;NUMA gets 128 words of noise at index NST
      DO 4 M=1,1024  ;apply hamming window to NUMA and fill FOURN with
C                       NUMA and the rest zeroes
        IF(M.GE.129) FOURN(M)=CMPLX(0.0,0.0)
        IF(M.GE.129) GO TO 4
        FOURN(M)=CMPLX(NUMA(M),0.0)*HAMMING(M)
 4      CONTINUE
      CALL DFT5(FOURN,1024,0)  ;calculate FFT of FOURN
      IF(IST.GE.(NUMTF+1)) GO TO 5  ;check for storage filled-up
      CALL WRBLK(1,((IST-1)*16),IO2,16,IE)  ;store FFT
      IF(IE.NE.1)TYPE"WRBLK1 ERROR"
      IF(IST.EQ.NUMTF) GO TO 9  ;check for enough FFTs calculated
      GO TO 3  ;go calculate and store more FFTs of noise
 5    DO 6 N=0,(NUMTF-2)  ;shift stored FFTs of noise
        CALL RDBLK(1,((N+1)*16),IO4,16,IE)
        IF(IE.NE.1)TYPE"RDBLK1 ERROR",IE
        CALL WRBLK(1,(N*16),IO4,16,IE)
        IF(IE.NE.1)TYPE"WRBLK2 ERROR",IE
 6      CONTINUE
 7    CALL WRBLK(1,((NUMTF-1)*16),IO2,16,IE) ;store new FFT in opened
C                                              space after shift
      IF(IE.NE.1)TYPE"WRBLK3 ERROR",IE
C
```

```
C**** CALCULATE POWER SPECTRUM ESTIMATE OF NOISE
C
        DO 8 I=1,1024  ;clear arrays involved in calculations
          SPEC(I)=CMPLX(0.0,0.0)
 8        FOURS(I)=SPEC(I)
 9      DO 11 M=0,(NANAL-1)  ;M represents reference frame
          DO 10 N=-7,3  ;N is the shift q around M
            NBLK=7+M+N  ;calculation of location of FFT of frame
C                         of shift (q+M)*32 samples
            MBLK=7+M  ;calculation of location of FFT of frame
C                       of shift M*32 samples
            CALL RDBLK(1,(NBLK*16),IO2,16,IE)  ;read FFT into FOURN
            IF(IE.NE.1)TYPE"RDBLK2 ERROR",IE
            DO 10 I=1,1024  ;calculate sum of products of FFTs over
C                             shift of N*32 for a value of M
              A=I*-2*3.14159*N/32.0  ;calculate phase shift correction for shift
 10           FOURS(I)=FOURS(I)+CONJG(CONJG(FOURN(I))*CMPLX(COS(A),SIN(A)))
            CALL WRBLK(1,((NUMTF+NANAL+M)*16),IO4,16,IE) ;store sum over N
            IF(IE.NE.1)TYPE"WRBLK4 ERROR",IE
            CALL RDBLK(1,(MBLK*16),IO2,16,IE)  ;read FFT of shift M*32
            IF(IE.NE.1)TYPE"RDBLK3 ERROR",IE
            DO 11 J=1,1024  ;calculate power spectrum from product of
C                             sum over N and FFT of shift M*32
C                             and add to power spectrum estimate
C                             calculated for previous shift (M-1)*32
              SPEC(J)=SPEC(J)+CONJG(FOURN(J))*FOURS(J)
 11           FOURS(I)=CMPLX(0.0,0.0)
C
C**** CALCULATION OF CROSS SPECTRUM BETWEEN NOISE AND SPEECH PLUS NOISE
C
        CALL SPCH(HAMMING,IST2,NANAL)
C
C**** CALCULATION OF WIENER FILTER AND FILTERING OF NOISE CHANNEL
C
        CALL RDBLK(1,112,IO4,16,IE)  ;read FFT of noise from frame M=0
        CALL CHECK(IE)
```

```
      DO 12 I=1,1024  ;calculate Weiner filter and multiply by FFT
C                         of noise (filtering)
        SPEC(I)=CONJG(FOURS(I))/1024*FOURN(I)/SPEC(I)
 12     FOURS(I)=CMPLX(0.0,0.0)
C
C**** RECONSTRUCTION OF CORRECTION SIGNAL BY OVERLAP AND ADD METHOD
C
      CALL DFT5(SPEC,1024,1)  ;inverse fourier transforming filtered noise
      IF(ICONT.NE.1025)CALL VFETCH(NUMA,ICONT-1,4) ;fetch previous samples
C                                                   of correction signal
      DO 13 I=1,128  ;add present values to past values of correction signal
C                       with an overlap of 128-32=96 samples
        IF(ICONT.EQ.1025)NUMA(I+32)=0  ;if first time past samples =0
 13     NUMA(I)=REAL(SPEC(I))*.46296+NUMA(I+32) ;.46296=scaling factor
      CALL VSTASH(NUMA,ICONT,4)  ;store new values of correction signal in
C                                  extended memory at index ICONT
      CALL CHECK(IE)
      ICONT=ICONT+1  ;increment index of ext. memory
      IF(ICONT.LE.1273)GO TO 15  ;check for 31 blocks of signal generated
      IF(IST.GE.380)GO TO 15  ;precaution to insure the next sequence is
C                               excecuted once
C
C**** OUTPUT OF CORRECTION SIGNAL TO NSFILT
C
      CALL EWRB(5,0,128,31,IE)  ;write to NSFILT first 31 blocks
      CALL CHECK(IE)
      ICONT=1026  ;reinitialize index
      DO 14 I=0,1  ;clear extended memory for new values of signal
 14     CALL VSTASH(IO4,1025+I*128,128)
      CALL VSTASH(NUMA,1025,4)  ;store present values of signal
 15   GO TO 3  ;go and start process loop over again
 16   CONTINUE
      CALL EWRB(5,31,128,30,IE)  ;write final 30 blocks to NSFILT
      CALL CHECK(IE)
      CALL RESET  ;close channels
      END
```

```
C
C**** THIS SUBROUTINE CALCULATES THE CROSS SPECTRUM BETWEEN SPEECH
C**** PLUS NOISE AND NOISE CHANNELS
C
C**** SET UP AND INITIALIZE ARRAYS
C
        SUBROUTINE SPCH(HAMMING, IST2, NANAL)
        DIMENSION NUMB(128)
        COMPLEX FOURN(1024),SPEC(1024),FOURS(1024),HAMMING(128)
        COMMON /BLKO/IO1(1024)/BLK/IO2(4096),IO4(4096)
        EQUIVALENCE(FOURS(1),IO4(1))
        EQUIVALENCE(FOURN(1),IO2(1))
        DO 100 I=1,1024  ;clear arrays
          SPEC(I)=CMPLX(0.0,0.0)
          FOURN(I)=CMPLX(0.0,0.0)
          FOURS(I)=CMPLX(0.0,0.0)
100     CONTINUE
        NUMTF=NANAL+10  ;calculate # of FFTs used in power spectrum estimates
C
C**** START CALCULATING AND STORING FFTS
C
 63     IST2=IST2+1  ;increment index
        NST=IST2+535  ;offset of 535 needed to read from ext. mem.
C                      block 66 and on
        CALL VFETCH(NUMB,NST,4)  ;get 128 samples of speech plus noise at NST
        DO 68 M=1,1024  ;apply Hamming window to NUMB and fill FOURN with
C                        NUMB and the rest zeroes
          IF(M.GE.129)FOURN(M)=CMPLX(0.0,0.0)
          IF(M.GE.129)GO TO 68
          FOURN(M)=CMPLX(NUMB(M),0.0)*HAMMING(M)
68        CONTINUE
        CALL DFT5(FOURN,1024,0)  ;calculate FFT of FOURN
        IF(IST2.GE.(NANAL+1)) GO TO 71  ;check for storage filled-up
        CALL WRBLK(1,((NUMTF+IST2-1)*16),IO2,16,IE)  ;store FFT
        IF(IE.NE.1)TYPE"WRBLK2 ERROR2 ",IE
        IF(IST2.EQ.NANAL) GO TO 73  ;check for enough FFTs calculated
```

```
      GO TO 63  ;if not go calculate FFT of another frame
 71   DO 72 N=0,(NANAL-2)  ;shift stored FFTs on disk
        CALL RDBLK(1,((NUMTF+N+1)*16),IO4,16,IE)
        IF(IE.NE.1)TYPE"RDBLK4 ERROR2 ",IE
        CALL WRBLK(1,((NUMTF+N)*16),IO4,16,IE)
 72     IF(IE.NE.1)TYPE"RDBLK5 ERROR2 ",IE
      CALL WRBLK(1,((NUMTF+NANAL-1)*16),IO2,16,IE)  ;write FFT in opened
C                                                     space after shift
      IF(IE.NE.1)TYPE"RDBLK6 ERROR2 ",IE
C
C**** CALCULATE CROSS POWER SPECTRUM ESTIMATE
C
 73   DO 107 M=0,(NANAL-1)  ;M represents reference frame
      CALL RDBLK(1,((NUMTF+NANAL+M)*16),IO2,16,IE)  ;read sum of products
      IF(IE.NE.1)TYPE"ERROR ON IO2 RDBLK ",IE
      CALL RDBLK(1,((M+NUMTF)*16),IO4,16,IE)  ;read reference frame
      IF(IE.NE.1)TYPE"ERROR ON IO4 RDBLK",IE
        DO 110 J=1,1024  ;calculate power spectrum estimate from product of
C                         sum over N (calculated in main program) and
C                         FFT of frame M of speech plus noise and add to
C                         previous estimate
          SPEC(J)=SPEC(J)+CONJG(FOURS(J))*FOURN(J)
          IF(M.NE.(NANAL-1))GO TO 110  ;check for last estimation
          FOURN(J)=SPEC(J)  ;FOURN gets estimate and interfaces to main
C                            program through labeled common
          SPEC(J)=CMPLX(0.0,0.0)
110       CONTINUE
107   CONTINUE
      RETURN
      END
```

```
C       **********************************************************
C       **********************************************************
C       **                                                      **
C       **      PROGRAM:   SUBTRACT                             **
C       **                                                      **
C       **              AUTHOR: LT CHRIS BATCHELOR              **
C       **                                                      **
C       **                      DATE:  DEC.  1,1981             **
C       **                                                      **
C       **********************************************************
C       **********************************************************
C
C       THIS PROGRAM SUBTRACTS THE NOISE ESTIMATE FROM SPEECH PLUS
C       NOISE TO COMPLEETE NOISE CANCELLATION.
C
C**** START PROGRAM, SET UP ARRAYS
C
        DIMENSION NSPFIL(7),NOUTFIL(7),NSPEECH(2048),NOISE(768),NADD(768)
1       FORMAT(S13)
        CALL DFILW("PSPEC",IE)  ;delete FFT storage file
        OPEN 4,"NAME"  ;open file containing names of I/O files
        READ(4,1) NSPFIL(1)  ;read filename of speech plus noise
        READ(4,1) NOUTFIL(1)  ;move pointer
        READ(4,1) NOUTFIL(1)  ;read output filename
        OPEN 2,NSPFIL  ;open speech plus noise file
        OPEN 1,NOUTFIL,ATT="BC",REC=88  ;create and open output file
        OPEN 3,"NSFILT"  ;open correction signal file
C
C**** START SUBTRACTION
C
        IST=0
        NBLK=61
 100    CALL RDBLK(2,IST,NSPEECH,2,IE)  ;read 2 blocks of speech plus noise
        IF(IE.NE.1)TYPE"RDBLK NSPEECH ERROR ",IE
        CALL RDBLK(3,IST,NOISE,1,IE)  ;read 1 block of noise estimate
        IF(IE.NE.1)TYPE"ERROR IN NOISE RDBLK ",IE
```

```
      DO 2 I=1,256  ;subtract with shift of 224 on speech plus noise
C                   because noise cancellation begins then
2     NADD(I)=NSPEECH(I+224)-NOISE(I)
      CALL WRBLK(1,IST,NADD,1,IE)  ;write noise cancelled speech
      IF(IE.NE.1)TYPE"WRTBLK ERROR ",IE
      IST=IST+1  ;increment counter
      IF(IST.LE.NBLK) GO TO 100  ;is processing through
      CALL DFILW("NSFILT",IE)  ;delete correction signal file
      CALL RESET
      END
```

Noise Generation

The following source listing for NOISE explains the noise generation process.

```
C       ************************************************************
C       ************************************************************
C       **                                                        **
C       **      PROGRAM:  NOISE                                   **
C       **                                                        **
C       **              AUTHOR: LT CHRIS BATCHELOR                **
C       **                                                        **
C       **                      DATE:  DEC. 1, 1981               **
C       **                                                        **
C       ************************************************************
C       ************************************************************
C
C
C
C       THIS PROGRAM GENERATES 62 DISK BLOCKS OF PSUEDO RANDOM
C       GAUSSIAN WHITE NOISE.  FIRST, UNIFORM RANDOM NUMBERS FROM
C       1 TO ((2**32)-1) BY REMAINDERING.  THE UNIFORM RANDOM DEVIATE
C       IS EQUAL TO THE REMAINDER OF THE LAST DEVIATE GENERATED TIMES 16607
C       DIVIDED BY THE MERSEENE PRIME ((2**32)-1).  THIS DEVIATE IS DIVIDED
C       BY ((2**32)-1) TO CONVERT THE RANGE OF DEVIATES TO 0-1.  NEXT,
C       THIS UNIFORM DEVIATE IS CONVERTED TO ONE OF A GAUSSIAN PROBABILITY
C       DISTRIBUTION WITH MEAN = 0 AND VARIANCE = VAR (INPUTED).  THIS IS
C       ACCOMPLISHED BY FORCING THE CUMULATIVE PROBABILITY OF THE UNIFORM
C       DEVIATE TO BE EQUAL TO THE CUMULATIVE PROBABILITY OF THE GAUSSIAN
C       DEVIATE.  INTEGRATION BY SIMPSON'S RULE OF THE GAUSSIAN DISTRIBUTION
C       FROM THE SMALLEST NEGATIVE NUMBER OF THE COMPUTER UP TO THE
C       GAUSSIAN DEVIATE THAT MAKES THE INTEGRATION EQUAL TO THE UNIFORM
C       DEVIATE (= CUMULATIVE PROBABILITY OF UNIFORM DEVIATE) IS THE METHOD
C       THAT TRANSFORMS RANDOM DEVIATES.
C
C*******************************************************************************
C
C**** START PROGRAM
C
        DOUBLE PRECISION A,IX,M,IY
        INTEGER XOUT(15872)  ;array containing noise for output
```

```
      ACCEPT "ENTER SEQUENCE LENGTH ",NSQ
      ACCEPT "VARIANCE (0-2047) = ",VAR
      CALL OPEN(12,"NUM3",3,IERR)  ;open NUM3  to accept output noise
      TYPE "ERROR CODE = ",IERR
C
C**** INITIALIZE ARRAY AND VARIABLES
C
      A=16607.D0
      DO 3 I=1,15872  ;clear XOUT
 3     XOUT(I)=0
      PI=3.1415927
      M=2147483647.D0  ;Merseene prime
      DELTA=1.0
      IY=1.D0  ;seed value for starting generator
      CON=SQRT(2.0*PI*VAR)  ;constant of gaussian distribution
      CON1=-1.0*ALOG(.72E-76*CON)
C
C**** START GENERATOR LOOP
C
      DO 10 I=1,NSQ
       IX=IY  ;IX gets previous uniform deviate or seed (I=1)
       IX=A*IX  ;A=16607
       IX=DMOD(IX,M)  ;find remainder of IX/M (M=((2**32)-1))
       IY=IX  ;IY is set up with uniform deviate
       IX=IX/M  ;limit range of deviates to 0-1
       XX=IX  ;double precision to single precision
       XNEW=0.0
       T=-1.0*SQRT(CON1*2.0*VAR)  ;lowest value of gaussian deviates
       IT=IFIX(T)
       T=FLOAT(IT)  ;get rid of fractional part
       ILIM=-2*IT  ;the full range of gaussian deviates
       XOLD=0.72E-76  ;the first gaussian deviate
       DO 13 J=1,ILIM  ;start integration but no farther tha ILIM
        T=T+DELTA  ;increment gaussian deviate
        XBEF=(1.0/CON)*EXP(-T*T/VAR/2.0)  ;calculate probability of gaussian
c                                            deviate
```

```
            XNEW=XNEW+((XOLD+XBEF)/2.0)  ;calculate cumulative probability of
C                                         gaussian deviate
            XOLD=XBEF  ;old=new deviate
            IF(XNEW-XX)13,13,7  ;is cumulative probability of gaussian
C                                 deviate equal to that of the uniform
  13        CONTINUE
C
C**** OUTPUT NOISE
C
  7        XOUT(I)=IFIX(T)  ;output gets gaussian deviate
  10       CONTINUE
          CALL WRBLK(12,0,XOUT,62,IERR)
          TYPE "IERR= ",IERR
          CALL RESET
          STOP
          END
```

Adding Noise to Speech

The following source listing for SPLUSN explains the process of adding noise to speech with a predefined signal to noise ratio.

```
C        ************************************************************
C        ************************************************************
C        **                                                        **
C        **      PROGRAM:   SPLUSN                                 **
C        **                                                        **
C        **              AUTHOR: LT CHRIS BATCHELOR                **
C        **                                                        **
C        **                      DATE:  NOV 2, 1981                **
C        **                                                        **
C        ************************************************************
C        ************************************************************
C
C*************************************************************************
C
C        THIS PROGRAM ADDS NOISE TO SPEECH TO OBTAIN A NOISE CORRUPTED
C        SPEECH SIGNAL WITH A DESIRED SIGNAL TO NOISE RATIO.  THE
C        ENERGY IN BOTH SPEECH AND NOISE IS DETERMINED BY CALCULATING
C        THE DFT OF 256 POINT SEGMENTS AND TAKING THE SUM OF THE MAGNITUDES
C        SQUARED OF COMPONENTS 1-128.  THE MAGNITUDE SPECTRA
C        CAN BE WEIGHTED BY A B WEIGHTING FUNCTION IF DESIRED.
C
C****************************************************************************
C
C**** START PROGRAM AND SET UP ARRAYS
C
         DIMENSION NSPFIL(7),NOISFIL(7),NSPEECH(256),NOISE(256),NOUT(256)
         DIMENSION B(128)  ;set up B weighting array
         COMPLEX FOURN(256),FOURS(256)
         INTEGER OUTFILE(7)
C
C**** GET FILENAMES AND DATA INFORMATION
C
1        FORMAT(S13)
         TYPE" PROGRAM ADDS SPEECH AND NOISE, "
         ACCEPT" SPEECH FILENAME ? "
         READ(11,1) NSPFIL(1)  ;read speech filename
```

```
      ACCEPT" NOISE FILENAME ? "
      READ(11,1) NOISFIL(1)  ;read noise filename
      ACCEPT" INPUT SIGNAL TO NOISE RATIO (DB) ",SNR
      ACCEPT" DESIRE B WEIGHTING ON NOISE SPECTRA ? Y=1,N=0 ",NB
      ACCEPT" DESIRE B WEIGHTING ON SPEECH SPECTRA ? Y=1,N=0 ",NSB
      ACCEPT" OUTPUT FILENAME ? "
      READ(11,1) OUTFILE(1)  ;read output filename
C
C**** OPEN I/O FILES
C
      CALL OPEN(2,NSPFIL,2,IE)
      IF(IE.NE.1)TYPE"NSPCFIL OPEN ERROR ",IE
      CALL OPEN(3,NOISFIL,2,IE)
      IF(IE.NE.1)TYPE"NOISFIL OPEN ERROR ",IE
      CALL OPEN(1,OUTFILE,2,IE)
      IF(IE.NE.1)TYPE"OUTFILE OPEN ERROR ",IE
C
C**** CALCULATE ENERGY IN SPEECH AND NOISE
C
      SPOWER=0.0
      ENNOISE=0.0
      KN=1
      KS=1
      DO 50 I=1,128  ;initialize B weighting array
        K=I-1
 50     B(I)=7160.0*7160.0*K*K/(K**4+4.90256E7*K*K+1.2544E12)
      DO 110 I=0,87
        CALL RDBLK(2,I,NSPEECH,1,IE)  ;read a block of speech
        CALL RDBLK(3,I,NOISE,1,IE)  ;read a block of noise
          DO 101 IM=1,256  ;initialize complex arrays with sampled data
            FOURS(IM)=CMPLX(NOISE(IM),0.0)
 101        FOURN(IM)=CMPLX(NSPEECH(IM),0.0)
        CALL DFT5(FOURN,256,0)  ;calculate DFT of speech
        CALL DFT5(FOURS,256,0)  ;calculate DFT of noise
        DO 103 IF=1,128  ;calculate energy in speech and noise
          IF(NB.EQ.1)KN=B(IF)
```

```
           IF(NSB.EQ.1)KS=B(IF)
           ENNOISE=ENNOISE+(CABS(FOURS(IF))**2)*KN/2.8836E6
103        SPOWER=SPOWER+(CABS(FOURN(IF))**2)*KS/2.8836E6
110     CONTINUE
        TYPE"SPEECH POWER= ",SPOWER
        CONSTANT=SQRT(SPOWER/ENNOISE/10**(SNR/10))  ;calculate gain applied to
C                                                     noise before adding to
C                                                     speech for SNR
        TYPE"CONSTANT= ",CONSTANT
C
C**** ADD NOISE TO SPEECH FOR SNR AND OUTPUT SPEECH PLUS NOISE
C
        DO 120 I=0,87
          CALL RDBLK(3,I,NOISE,1,IE)  ;read block of noise
          CALL RDBLK(2,I,NSPEECH,1,IE)  ;read block of speech
          DO 119 J=1,256  ;add noise to speech after gain is applyed to noise
 119      NOUT(J)=NSPEECH(J)+IFIX(CONSTANT*FLOAT(NOISE(J)))
          CALL WRBLK(1,I,NOUT,1,IE)
 120    CONTINUE
        TYPE"NOISE POWER= ",ENNOISE
        END
```

APPENDIX B

The Use of the ILS System

The use of the ILS system to perform linear predictive analysis/synthesis of a speech file is done by executing the following steps.

1. 2ILSFIL - The system asks you for the name of CHOPS file, the name of ILS file to be created (WD#). # is a file number desired.
2. FIL WD# - Necessary to designate WD# as primary file.
3. INA - Initializes the LPC analysis requirements. The system will ask you for the values of these requirements.
4. API N1,N2 - Performs the LPC analysis from frame N1 to N2. N1 must be greater than or equal to 3. The API command takes speech information from the primary file WD# and stores the analysis parameters in a secondary file.
5. FIL U# - Unprotects file WD# so that the synthesis program can take the synthesized speech and store it back into the primary file WD#.

6. SNS - Performs synthesis of speech from parameters stored in secondary file.

7. AGC - Program which multiplies resulting speech file by a gain factor to prevent clipping when speech is outputted through the D/A converter.

Vita

Christopher Lee Batchelor was born on 22 December 1954 in Corona CA. He graduated from Havelock High School in Havelock NC in 1972. From August of 1973 to May of 1977, he attended North Carolina State University in Raleigh NC. There he received, as an Honor Graduate, the degree of Bachelor of Science in Electrical Engineering in May 1977. He then worked as Assistant Engineer for AAI, Inc. in Cockeysville MD with the duties of radar test set development. In August 1978 he went to work as Assistant Engineer for Babcock and Wilcox in Lynchburg VA with the duties of microprocessor based control system development. Next, in January 1980 he attended Officers' Training School and received a commission in the United States Air Force in June 1980. He then entered the School of Engineering of the Air Force Institute of Technology.

Permanent Address: 506 Macon Street
Wilson NC 27893

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFIT/GE/EE/81D-10
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): NOISE CANCELLATION TO IMPROVE THE QUALITY OF LPC PROCESSED SPEECH DEGRADED BY NOISE
5. TYPE OF REPORT & PERIOD COVERED: MS Thesis
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Christopher L. Batchelor, 2LT USAF
8. CONTRACT OR GRANT NUMBER(s):
9. PERFORMING ORGANIZATION NAME AND ADDRESS: AF Institute of Technology (AFIT-EN), Wright-Patterson AFB OH 45433
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: Electrical Engineering Department, Air Force Institute of Technology, Wright-Patterson AFB OH 45433
12. REPORT DATE: December 1981
13. NUMBER OF PAGES: 72
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

15 APR 1982

Lynn E. Wolaver

Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

18. SUPPLEMENTARY NOTES

Approved for Public Release IAW AFR 190-17

~~FREDERICK C. LYNCH, Major,~~ USAF
Director of Public Affairs

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

LPC
Linear Predictive Coding
Speech Processing
Digital Signal Processing
Power Spectrum Estimation
Noise Cancellation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Methods for improving the quality of the speech resulting from linear predictive analysis/synthesis of speech degraded by background noise are discussed. A method of noise cancellation using Wiener filtering in the frequency domain with the short-time Fourier transform was chosen for implementation. Implementation was done on a Data General Nova/Eclipse digital signal processing system in FORTRAN 5. Speech degraded by white gaussian noise was processed through linear predictive analysis/synthesis with and without noise cancellation

(Continued on Reverse)

DD FORM 1473, 1 JAN 73 — EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

BLOCK 20: Abstract (Cont'd):

preprocessing. Preliminary laboratory listenings verified that an improvement in quality was achieved with noise cancellation preprocessing. Although improvement in quality was achieved, more effort is required to make this implementation more efficient and improve the quality of speech produced.

ILMED

8